PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2000-333128
(43)Date of publication of application : 30.11.2000

(51)Int.Cl. H04N 5/92
H04N 7/24

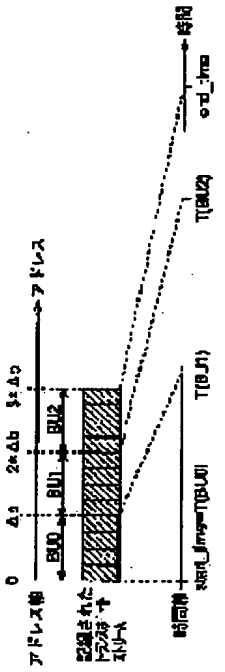
(21)Application number : 11-137203
(22)Date of filing : 18.05.1999

(71)Applicant : SONY CORP
(72)Inventor : KATO MOTOKI
HAMADA TOSHIYA
NAKAMURA MASANOBU

(30)Priority
Priority number : 11072303 Priority date : 17.03.1999 Priority country : JP

(54) DATA PROCESSOR, ITS METHOD AND MEDIUM

(57)Abstract:
PROBLEM TO BE SOLVED: To efficiently record a coding stream onto a recording medium and to quickly apply random access to the recording medium.
SOLUTION: Packets of a prescribed number of a channel whose recording is designated from a received transport stream are collected, divided into block units and recorded in a recording medium without adding a dummy packet. A data length of a block unit BU0 expressed by a value subtracting a time T(BU0) at a head of a time unit BU0 from a time T(BU1) at a head of a time unit BU1, for example is cross-referenced with an address block-unit-address at a head of recorded packets among packets of each of block units BU0, BU1, BU2 and the result is recorded in the recording medium as a block unit map.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

* NOTICES *

CLAIMS

[Claim(s)]
[Claim 1] In the data processor which processes the data contained in the inputted coding stream The unitization means which carries out unitization of said inputted coding stream to the unit of a predetermined unit, A storage means to memorize the coding stream by which unitization was carried out with said unitization means, The 1st creation means which creates a unit map including the amount-of-data information showing the amount of the data for every unit of said coding stream by which unitization was carried out with said unitization means, The data processor characterized by having the 2nd creation means which shows the location of the entry point for every program of said coding stream, and which creates the entry point map subordinate to said unit map.
[Claim 2] Said amount-of-data information on said unit map is a data processor according to claim 1 characterized by what is expressed with the address of said storage means.
[Claim 3] Said amount-of-data information on said unit map is a data processor according to claim 1 characterized by what is expressed with the time amount corresponding to the amount of data of said unit memorized by said storage means.
[Claim 4] Said 2nd creation means is a data processor according to claim 1 characterized by changing said entry point map when said coding stream is edited.
[Claim 5] The data processor according to claim 1 characterized by having further a file-ized means to file-ize either [at least] said unit map or an entry point map with said coding stream.
[Claim 6] The data processor according to claim 5 characterized by having further a record means to record the data file-ized by said file-ized means on a record medium.
[Claim 7] Said 1st creation means is a data processor according to claim 1 characterized by changing said unit map when said coding stream is edited.
[Claim 8] In the data-processing approach of the data processor which processes the data contained in the inputted coding stream The unitization step which carries out unitization of said inputted coding stream to the unit of a predetermined unit, The storage step which memorizes the coding stream in which unitization was carried out by processing of said unitization step, The creation step which creates a unit map including the amount-of-data information showing the amount of the data for every unit of said coding stream in which unitization was carried out by processing of said unitization step, The data-processing approach characterized by including the 2nd creation step which creates the entry point map subordinate to said unit map in which the location of the entry point for every program of said coding stream is shown.
[Claim 9] The unitization step which carries out unitization of said coding stream which is the program which processes the data contained in the inputted coding stream, and was inputted to the unit of a predetermined unit, The storage step which memorizes the coding stream in which unitization was carried out by processing of said unitization step, The creation step which creates a unit map including the amount-of-data information showing the amount of the data for every unit of said coding stream in which unitization was carried out by processing of said unitization step, The medium which makes a computer execute the program characterized by including the 2nd creation step which creates the entry point map subordinate to said unit map in which the location of the entry point for every program of said coding stream is shown.

[Translation done.]

* NOTICES *

DETAILED DESCRIPTION

[Detailed Description of the Invention]
[0001]
[Field of the Invention] By extracting focus information from a coding stream about a medium in a data processor and an approach, and a list, especially this invention relates to a medium at the data processor which could be made to do random access quickly and an approach, and a list, when two or more programs are multiplexed.
[0002]
[Description of the Prior Art] In multi-channel digital television broadcast of DVB (Digital Video Broadcast) of Europe, digital BS broadcast of Japan, etc., an MPEG(Moving Picture Experts Group) 2 transport stream is used. A transport stream is a stream which the transport packet followed, and, as for a transport packet, for example, an MPEG 2 video stream and an MPEG1 audio stream are packet-ized. One or more AV (Audio Visual) programs are multiplexed by one transport stream transmitted by the electric wave of broadcast. Generally, it is carrying out mutually-independent [of the AV program of each channel].
[0003] Therefore, if it receives as it is and the transport stream sent by broadcast is recorded with a domestic receiver, the program of all the channels of the transport stream is recordable on coincidence. Moreover, if what separated the transport stream of AV program of some channels chosen by the user from the transport streams sent by broadcast is recorded, the program of a selected number of arbitration of channels is recordable on coincidence.
[0004] The example of the record approach of the conventional transport stream is shown in drawing 1 . Drawing 1 (A) shows the transport stream by which two or more AV programs were multiplexed. An axis of abscissa is time amount and is divided into every [of spacing of deltat] block unit BUi (i= 0, 1, 2 ...) here. When one or more AV programs are chosen out of an input transport stream, the selected transport packet gives a slash and is shown. As the selected transport packet is generally shown in drawing 1 (B), it appears to irregular timing and the number of the transport packets for every block unit BUi changes.
[0005] As shown in drawing 2 , the transport packet as which it was chosen for every block unit BUi of spacing of deltat packs spacing, and is recorded on a record medium. At this time, each transport packet adds the time stump in which the time of day on each stream is shown, and is recorded. This time stump is the same as that of TSP_extra_header of 4-byte length added to the transport packet specified for example, in DV (Digital Video) format.

[0006] In drawing 2 , an axis of abscissa is the address which shows the byte position of the recorded transport stream. If the transport stream of a Variable Bit Rate as shown in drawing 1 (B) is inputted, as shown in drawing 2 , a recording apparatus will put in dummy data and will record data at the record rate of immobilization. Therefore, the amount of data of the file to the passage of time of the recorded transport stream is proportional. That is, if the record amount of data per block unit is set to x, the byte position of the initial data of the n-th block unit (n= 0, 1, 2, ...) will be set to x n times as many as this.
[0007]
[Problem(s) to be Solved by the Invention] Thus, since the conventional record approach inserts dummy data and makes it the fixed record rate, its recording efficiency of a transport stream is not good. However, if dummy data is not inserted, since the passage of time of a transport stream and the amount of data of a file which were recorded stop being proportional, when accessing the data of the position on the time-axis of a transport stream, the problem to which the access nature of data worsens occurs.
[0008] Moreover, generally, by the stream of MPEG 2 video, I picture is encoded at intervals of about 0.5 seconds, and the other picture is encoded as P picture or a B picture. Therefore, I picture must be searched when carrying out high-speed playback of the video signal from the record medium with which the stream of MPEG 2 video was recorded. However, when reproducing by random access from the record medium with which transport streams, such as digital broadcast, were recorded, it was difficult to search the start byte of I picture efficiently. That is, the syntax of the video stream read from the random byte position of the transport stream on a record medium was analyzed, and the start byte of I picture or an audio frame was searched. Therefore, it was difficult for the search of I picture to take time amount depending on the case, and to carry out quick random access playback of a response to a user input.
[0009] This invention is made in view of such a situation, and when two or more programs are multiplexed, it can be made to do random access quickly.
[0010]

[Means for Solving the Problem] In the data processor which processes the data contained in the coding stream into which the data processor according to claim 1 was inputted The unitization means which carries out unitization of the inputted coding stream to the unit of a predetermined unit, A storage means to memorize the coding stream by which unitization was carried out with the unitization means, The 1st creation means which creates a unit map including the amount-of-data information showing the amount of the data for every unit of the coding stream by which unitization was carried out with the unitization means, It is characterized by having the 2nd creation means which shows the location of the entry point for every program of a coding stream and which creates the entry point map subordinate to a unit map.
[0011] The amount-of-data information on said unit map can be expressed with the time amount corresponding to the address of a storage means, or the amount of data of the unit memorized by the storage means.
[0012] When a coding stream is edited by said 2nd creation means, an entry point map can be changed.
[0013] A record means to record the data file-ized by said file-ized means on a record medium can be established further.
[0014] A unit map can be changed when a coding stream is edited by said 1st creation means.
[0015] In the data-processing approach of the data processor which processes the data contained in the coding stream into which the data-processing approach according to claim 8 was inputted The unitization step which carries out unitization of the inputted coding stream to the unit of a predetermined unit, The storage step which memorizes the coding stream in which unitization was carried out by processing of a unitization step, The creation step which creates a unit map including the amount-of-data information showing the amount of the data for every unit of the coding stream in which unitization was carried out by processing of a unitization step, It is characterized by including the 2nd creation step which creates the entry point map subordinate to a unit map in which the location of the entry point for every program of a coding stream is shown.
[0016] The unitization step which carries out unitization of the coding stream which the program of a medium according to claim 9 is a program which processes the data contained in the inputted coding stream, and was inputted to the unit of a predetermined unit, The storage step which memorizes the coding stream in which unitization was carried out by processing of a unitization step, Unitization was carried out by processing of a unitization step. The creation step which creates a unit map including the amount-of-data information showing the amount of the data for every unit of a coding stream, It is characterized by including the 2nd creation step which creates the entry point map subordinate to a unit map in which the location of the entry point for every program of a coding stream is shown.
[0017] In a data processor according to claim 1, the data-processing approach according to claim 8, and a medium according to claim 9, unitization of the coding stream is carried out and a unit map including the amount-of-data information showing the amount of the data for every unit is created.
[0018]
[Embodiment of the Invention] Although this invention is explained by making into an example the case where it is the multiplexing stream by which the program whose coding stream is one or more is multiplexed hereafter, this invention is applicable even if coding streams are elementary streams, such as an MPEG video stream.
[0019] First, the fundamental principle of this invention is explained. The dynamic-image recording device of this invention calculates the time of day on a break and the stream of the data for every block unit for every block unit of the predetermined amount of data, when recording the transport stream by which one or more programs are multiplexed on record media, such as a disk and a tape. And the block unit map in which the time of day on the stream of the data for every block unit of this is shown is created. Furthermore, the entry point map in which the location of the entry point (random access point) for every program of a transport stream to record is shown is created. An entry point map has the structure subordinate to a block unit map. This block unit map is explained below.
[0020] Drawing 3 shows the transport stream by which two or more AV programs were multiplexed. An axis of abscissa is a time-axis which shows the time of day on a stream here. It is chosen out of an input transport stream for record of one or more AV programs. The selected transport packet attaches a slash and is shown. The selected transport packet appears to irregular timing, as generally shown in drawing 3 (B).
[0021] As shown in drawing 4 , the selected transport packet packs spacing and is recorded on a record medium. At this time, the time stump in which the time of day on each stream is shown is added to each transport packet. Let this time stump be the same thing as TSP_extra_header of 4-byte length added to the transport packet specified by the DV format.
[0022] The time of day on a stream is expressed as the address by which the byte position of the recorded transport stream is shown on an axis of abscissa to drawing 4 . The recorded transport stream is divided for every block unit of predetermined amount-of-data deltab. deltab is made into the amount of data for four transport packets in this example. A block unit is indicated to be BU. The figure which continues behind BU shows the time order foreword of BU. At the time of the first original record, all the data lengths of BU are the same and it is deltab. In addition, the data length of deltab is good also as an integral multiple of for example, not only this example but an ECC block size.
[0023] The time-axis of drawing 4 shows the time-axis which shows the time of day on a transport stream. On this shaft, the time of day of the initial data for every block unit is shown. The time of day of the initial data of the block unit BU0, BU1, and BU2 is T (BU0), T (BU1), and T (BU2), respectively.
[0024] Drawing 5 shows the table of the time of day on the stream of the initial data for every block unit of a block

unit map, i.e., the recorded transport stream. Here, block_unit_address is the address of the initial data of the block unit on the recorded stream. Moreover, block_unit_time expresses the time of day of the initial data of a block unit, and delta_block_unit_time expresses the time amount length for every block unit. delta_block_unit_time for every block unit is table-ized on a block unit map.
[0025] In the example of drawing 5 , address block_unit_address of the data of the head of the block unit BU0 is set to 0, and corresponding time-of-day block_unit_time is set to T (BU0). Similarly, data-address block_unit_address of the head of the block unit BU1 is set to deltab, and corresponding time-of-day block_unit_time is set to T (BU1). Furthermore, address block_unit_address of the data of the head of the block unit BU2 is 2xdeltab Carried out, and corresponding time-of-day block_unit_time is set to T (BU2).
[0026] And time amount length delta_block_unit_time of the block unit BU0 is made into the difference (T(BU1)-T (BU0)) of the time of day T of the data of the head of the block unit BU1 (BU1), and the time of day T of the data of the head of the block unit BU0 (BU0). Time amount length delta_block_unit_time of the block unit BU1 is made into the difference (T(BU2)-T (BU1)) of the time of day T of the data of the head of the block unit BU2 (BU2), and the time of day T of the data of the head of the block unit BU1 (BU1). Furthermore, the time amount length (delta_block_unit_time) of the block unit BU2 is made into the difference (end_time-T (BU2)) with the time of day T of the data of time-of-day end_time of the data of the last of the block unit BU2, and the head of the block unit BU2 (BU2).
[0027] Next, an above-mentioned entry point map is explained. The transport streams shown in drawing 6 are the transport stream shown in drawing 4 , and same transport stream. The entry point shall have begun in the transport packet shown with a slash here. Specifically in an entry point, the sequence header and I picture data of MPEG video shall have begun. When an entry point exists in a predetermined block unit, the offset address from the start address of the data of the block unit to the address of an entry point is calculated.
[0028] That is, in the example of drawing 6 , an entry point (I picture) exists in the block units BU0 and BU2. Then, in the block unit BU0, the spacing a from the address 0 of the head to address I_start_address of the head of I picture is calculated as an offset address. In the block unit BU2, the spacing b from address 2xdeltab of the head to address I_start_address of the head of I picture is calculated as an offset address by coincidence.
[0029] Drawing 7 shows the example of an entry point map, i.e., the table of the offset address to the entry point for every block unit. entry_point_flag is set to "1" when an entry point exists in the corresponding block unit BUi, and when it does not exist, it is set to "0." About the block unit whose entry_point_flag is "1", offset address I_start_offset_from_block_unit_address from start-address block_unit_address of the data of the block unit to address I_start_address of an entry point is calculated, as shown in a degree type.
[0030]
I_start_offset_from_block_unit_address= I_start_address - For every block_unit_address and entry point, ending-address P2_end_address of the next next P of the next P of ending-address I_end_address of I picture data of an entry point and I picture of an entry point or ending-address P1_end_address of I picture, and I picture of an entry point or I picture is calculated, as shown in a degree type.
[0031] I_end_offset_address = I_end_address-I_start_addressP1_end_offset_address= P1_end_address-I_start_addressP2_end_offset_address= P2_end_address-I_start_address — the example of these addresses is shown in drawing 8 . Drawing 8 shows the MPEG video data which begins from the head of a predetermined block unit. Here, I, P, and B express I picture, P picture, or B picture, respectively, and the figure of a suffix shows the display order of a picture. I picture of the entry point shown by I2 exists in this block unit. Moreover, the next P picture of the I picture I2 is P5, and the next next P picture of the I picture I2 is P8. At this time, I_start_offset_from_block_unit_address calculated by the above-mentioned formula, I_end_offset_address, P1_end_offset_address, and P2_end_offset_address become the relation shown in drawing.
[0032] That is, let I_end_offset_address be the value which subtracted starting address I_start_address of the I picture I2 from ending-address I_end_address of the I picture I2. Let P1_end_offset_address be the value which subtracted starting address I_start_address of the I picture I2 from ending-address P1_end_address of the P picture P5. Furthermore, let P2_end_offset_address be the value which subtracted starting address I_start_address of the I picture I2 from ending-address P2_end_address of the P picture P8.
[0033] I_start_offset_from_block_unit_address is the address I of an entry point. From start_address, it considers as the value which subtracted address block_unit_address of the head of the data of a block unit.
[0034] In addition, when two or more programs are included in the transport stream to record, the information on an entry point is distinguished and created for every program. Moreover, it has the information (parsed_program_flag) which shows whether, as for an entry point map, entry point data exist for every program in consideration of the case where entry point data cannot be prepared, about all programs.
[0035] When the transport stream recorded on the record medium is edited, the block unit map is changed (updating). Next, the approach is explained. Drawing 9 (A) shows the example in the case of eliminating two packets of the head of a transport stream shown in drawing 4 . Drawing 9 (B) shows the transport stream after doing in this way and carrying out partial elimination of the packet. Drawing 10 shows the block unit map of the transport stream of drawing 9 (B). Thus, since the time amount length (first_block_unit_size) of the first block unit BU0 changes when the data to the middle of a block unit are eliminated, this is rewritten. the case of drawing 9 (B) — data length delta_block_unit_time of the block unit BU0 — the difference of the time stump of the packet x of start-address T (BU1) of the block unit BU1, and the head of the block unit BU0 after elimination — it is changed into a value.
[0036] Next, the example of the syntax of an above-mentioned block unit map is shown in drawing 11 and drawing

12 . Drawing 11 and drawing 12 are the header unit (BlockUnitMapHeader()) of a block unit map, respectively. Data division (BlockUnitMapData()) It expresses. When recording a block unit map as a file, you may record by making a header unit and data division one file, and may record as a separate file. start_time of BlockUnitMapHeader(), and end_time The recording start time of day and record end time when the start time and end time of this block unit map being shown, for example, recording a certain transport stream, respectively are shown. first_block_unit_size shows the time amount length of the first block unit. block_unit_size The time amount length of the block unit of the 2nd henceforth is shown. number_of_block_unit_entries shows the number of the block units in a transport stream. A number of delta_block_unit_address (drawing 5) shown by number_of_block_unit_entries is written to BlockUnitMapData().

[0037] Moreover, the 1st example of the syntax of an above-mentioned entry point map is shown in drawing 13 thru/or drawing 15 . Drawing 13 is the header unit (EntryPointMapHeader()) of an entry point map. Expressing, drawing 14 expresses the data division (EntryPointMapData()) of an entry point map. Drawing 15 expresses the syntax of entry_point_data() of drawing 14 further. When recording an entry point map as a file, you may record by making a header unit and data division one file, and may record as a separate file.

[0038] number_of_programs of EntryPointMapHeader() of drawing 13 shows the number of programs in a transport stream. There is information which shows whether an entry map table exists about each program recorded on the 6th line from the 3rd line of this syntax. program_number of the 4th line is information which specifies a program (discernment), and is information currently written to corresponding PMT (Program Map Table) of a program. parsed_program_flag of the 5th line shows whether the entry point data of the program exist.

[0039] The information on PMT of each program to record follows the 10th line from the 8th line. MPEG 2_TS_program_map_section() is PMT which was extracted out of the transport stream to record and which is specified by MPEG 2 systems specification. Here, NUMBER_OF_ParsedPrograms is the number of the programs whose parsed_program_flag is "1." The sequence that data appear in the loop formation of NUMBER_OF_ParsedPrograms of the 8th line is parsed_program_flag at the loop formation of number_of_programs of the 3rd line. It is the sequence that program_number which is "1" appears.

[0040] The data of the entry point about each program to record are described by EntryPointMapData() of drawing 14 . The parameters of the entry point about one block unit are entry_point_flag and entry_point_data(). The contents of entry_point_data() about one block unit are entry_point_time_stamp, I_start_offset_from_block_unit_address, I_end_offset_address, and P1_end_offset_address, as shown in drawing 15 . It is P2_end_offset_address. Here, entry_point_time_stamp is calculated based on PTS (Presentation Time Stamp) of the time of day on the stream of the transport packet of an entry point, or I picture of an entry point. PTS is PES of MPEG 2 systems specification. It is the information added to the header of a packet.

[0041] Moreover, the 2nd example of the syntax of an above-mentioned entry point map is shown in drawing 16 . The configuration of EntryPointMapHeader() and entry_point_data() is the same as that of the case where it is shown in the 1st above-mentioned drawing 13 or above-mentioned drawing 15 in an example. This drawing 16 and drawing 14 are compared, and the data list direction of the entry point about each program differs from the 1st example of drawing 14 so that clearly.

[0042] Next, the 1st example of the condition which shows below, and the example of the data list of the entry map in the case of [each] the 2nd example are shown. Here, as shown in drawing 17 , three programs (program#1, program#2, program#3) shall be multiplexed in the transport stream, and every block unit BUi (i= 0, 1, 2, 3) shall have the entry point of each program. In this case, each parameter is as follows.

[0043]

number_of_block_unit_entries =4 number_of_programs=3 program_number=1:parsed_program_flag=1 program_number=2:parsed_program_flag=1 program_number=3:parsed_program_flag=1 NUMBER_OF_ParsedPrograms=3

drawing 18 shows the entry point map in the case of the 1st example (example of drawing 14). In this case, it becomes the form where the list of entry point data separated for every program. That is, since entry_point_data#1-1 thru/or entry_point_data#1-4 exist in the block unit BU0 thru/or each of BU3 as entry_point_data as EntryPointMapData of program#1 is shown in drawing 18 (A), entry_point_flag is set to "1", respectively.

[0044] In addition, entry_point_data#A-B expresses entry_point_data() about the Bth entry point of program_number=A.

[0045] As EntryPointMapData of program#2 is shown in drawing 18 (B), since entry_point_data does not exist, the entry_point_flag is set to "0" at the block unit BU1 and BU3. On the other hand, in the block unit BU0 and BU2, since entry_point_data#2-1 and entry_point_data#2-2 exist, respectively, the entry_point_flag is set to "1."

[0046] Furthermore, since entry_point_data does not exist, entry_point_flag is set to "0" at the block unit BU0 of EntryPointMapData of program#3, and BU2. Since entry_point_data#3-1 and entry_point_data#3-2 exist, respectively, the entry_point_flag is set to "1" at the block unit BU1 and BU3.

[0047] entry_point_data is described to be these entry_point_flag by EntryPointMapData.

[0048] Moreover, drawing 19 shows the entry point map in the case of the 2nd example (example of drawing 16).

[0049] In this case, it becomes the form where the entry point data of each program are located in a line with time order for every block unit, and the list of entry point data becomes one form. That is, in the block unit BU0, three program program#1 thru/or #3 are described and entry_point_flag and corresponding entry_point_data are described about each. In this example, since that entry_point_flag is set to "0" since entry_point_data does not exist, and entry_point_data#1-1 and #2-1 exist in program#3 about program#1 and #2, that entry_point_flag is set to "1."

[0050] the other block units BU1 thru/or BU3 — also setting — program#1 thru/or #3 — entry_point_flag and

entry_point_data are described about each.
[0051] Next, the example of a configuration of the dynamic-image recording device 1 which creates an above-mentioned table and is recorded on a record medium with a transport stream from the inputted transport stream is shown in drawing 20 .
[0052] One or more AV programs are multiplexed by the transport stream inputted from a terminal 10. The channel (service name) of AV program chosen by the user interface is inputted into a terminal 22. One or plural are sufficient as the number of channels chosen here.
[0053] The PID filter 11 takes out the transport packet of PID (Packet ID) specified by the stream analysis section 12 out of the inputted transport stream. PID is the sign of 13 bit length in the fixed position of the header of a transport packet, and expresses the type of the data currently stored in the payload of the transport packet. The PID filter 11 takes out first the transport packet of PAT (Program Association Table) which is PID=0x0000. PID of the transport packet of PMT (Program Map Table) of each program multiplexed by the transport stream is written to PAT. The transport packet of PAT outputted from the PID filter 11 is inputted into the stream analysis section 12.
[0054] A counter 24 carries out counting of the number of packets to the present packet from the head packet of the transport stream to record, and outputs the present packet number to the block unit map creation section 23 and the entry point map creation section 16.
[0055] The stream analysis section 12 extracts PCR from the transport packet which transmits PCR (Program Clock Reference), and outputs it to the PLL section 13. When [some] there is two or more PID of the transport packet which transmits PCR, PCR is extracted from the packet of one PID. Synchronizing with inputted PCR, the PLL section 13 generates a clock with a frequency of 27MHz, and outputs the clock to the time stump generating section 14.
[0056] The time stump generating section 14 counts the inputted clock, and generates the time stump corresponding to the counted value. This time stump will express the elapsed time after record of zero, then its transport stream for the time stump of a transport packet recorded first. This time stump is outputted to the stream analysis section 12, the time stump adjunct 15, and the block unit map creation section 23.
[0057] The time stump adjunct 15 outputs the transport packet which added the time stump in which the arrival time is shown to the transport packet inputted from the PID filter 11, and the time stump added to it to the file system section 17.
[0058] The block unit map creation section 23 creates an above-mentioned block unit map based on the time stump inputted as the packet number inputted from a counter 24 from the time stump generating section 14. The created block unit map is outputted to the entry point map creation section 16 and the file system section 17.
[0059] The stream analysis section 12 outputs the program information shown in the degree for every program to the entry point map creation section 16.
(1) program_number of a program (2) PID of the transport packet of PMT of a program (3) PID and stream_type (4) of a transport packet of video which constitute a program A program PID and stream_type (5) of a transport packet of an audio to constitute PID of PCR of a program — here, stream_type It is the contents currently written on PMT, and in the case of video, stream types, such as MPEG 2 / MPEG1, are expressed, and, in the case of an audio, stream types, such as MPEG1/AC-3, are expressed.
[0060] The stream analysis section 12 creates the entry point data of a stream to record again, and inputs them into the entry point map creation section 16. The contents of entry point data are shown in drawing 15 . In addition, since the stream analysis section 12 takes out PTS from an input stream when setting the time stump of an entry point to PTS of an entry point, it is not necessary to input into the stream analysis section 12 the time stump created by the time stump generating section 14.
[0061] The entry point map creation section 16 table-izes entry point data for every program, creates an above-mentioned entry point map, and outputs it to the file system section 17.
[0062] Next, the actuation is explained. If a transport stream is inputted from a terminal 10, the PID filter 11 will extract the transport packet containing PID which is PID=0x0000, and will output it to the stream analysis section 12. The stream analysis section 12 performs processing shown in the flow chart of drawing 21 at this time.
[0063] At step S11, the stream analysis section 12 will acquire PID of the transport packet of PMT of each program ordered through the terminal 22 from the PAT, if the transport packet of PID=0x0000 is received from the PID filter 11.
[0064] At step S12, the stream analysis section 12 sets PID of PMT of each program to the PID filter 11. The PID filter 11 will output it to the stream analysis section 12, if a transport packet with PID of these PMT is taken out.
[0065] At step S13, the stream analysis section 12 receives the transport packet of PMT from the PID filter 11. PID of the packet which is transmitting PID and PCR (Program Clock Reference) of the transport packet which has the video stream which constitutes the program, and an audio stream in a payload is written to PMT. The stream analysis section 12 acquires PID of the packet which is transmitting PID and PCR of the transport packet which has the video stream which constitutes each program chosen by the user interface, and an audio stream in a payload here.
[0066] At step S14, the stream analysis section 12 is PID of the transport packet which has the video stream which constitutes each program chosen by the user interface, and an audio stream in a payload. PID of the packet which is transmitting PCR is set to the PID filter 11.
[0067] In addition, when PID of the packet of a service information which transmits EPG (Electrical Program Guide) etc. beforehand is known, these PID is also set to the PID filter 11, and the packet of these PID is also outputted

from the PID filter 11.
[0068] Thus, the transport packet extracted with the PID filter 11 is supplied to a counter 24, the stream analysis section 12, and the time stump adjunct 15. A counter 24 carries out counting of the number of packets to the present packet from the packet of the head of a transport stream to record, and detects the present packet number. Detected current packet NO. is supplied to the block unit map creation section 23 and the entry point map creation section 16.
[0069] Moreover, the stream analysis section 12 extracts PCR from the transport packet inputted, and supplies it to the PLL section 13. Synchronizing with inputted PCR, the PLL section 13 generates a clock with a frequency of 27MHz, and supplies it to the time stump generating section 14.
[0070] The time stump generating section 14 counts the inputted clock, and generates the time stump corresponding to the counted value. The time stump adjunct 15 adds the time stump which the time stump generating section 14 which shows the arrival time generated to the transport packet inputted from the PID filter 11, and supplies it to it at the file system section 17.
[0071] The block unit map creation section 23 creates the block unit map to which the packet number inputted from a counter 24, block_unit_address for every [as shown in drawing 5 based on the time stump inputted from the time stump generating section 14] block unit, and delta_block_unit_time were made to correspond, and supplies it to the entry point map creation section 16 and the file system section 17.
[0072] The stream analysis section 12 supplies the program information mentioned above for every program to the entry point map creation section 16 again.
[0073] For this reason, the stream analysis section 12 performs analysis processing of an entry point as shown in drawing 22 and drawing 23 .
[0074] The stream analysis section 12 sets the stream_type to the PID filter 11 at step S31 with PID of the video of the program to record. Thereby, the packet of the specified video is supplied to the stream analysis section 12 from the PID filter 11.
[0075] At step S32, the stream analysis section 12 initializes the pointer vpp of a video packet, and is set to vpp=0. Pointer vpp expresses the sequence of the video packet of the above PID which is carrying out current processing.
[0076] The stream analysis section 12 increments the pointer vpp of a video packet at step S33 (for example, only 1 increases).
[0077] At step S34, the stream analysis section 12 investigates whether sequence_header_code (it is the sign of "0x000001B3" at 32 bit length) of MPEG video is contained in the stream in a payload. When sequence_header_code is not contained, processing returns to step S33.
[0078] When judged with sequence_header_code being contained in a payload at step S34, it progresses to step S35 and the stream analysis section 12 makes I_start_address the address of the packet (packet of the first I picture) containing sequence_header_code (drawing 8).
[0079] The stream analysis section 12 increments the pointer vpp of a video packet at step S36.
[0080] At step S37, the stream analysis section 12 investigates whether the data of the above-mentioned I picture were completed. When the data of I picture are not completed yet, processing returns to step S36. When the data of I picture are completed, processing progresses to step S38.
[0081] At step S38, the stream analysis section 12 makes the address of the packet which I picture ends I_end_address (drawing 8). It means that the address of the first I picture had been determined by the above.
[0082] The stream analysis section 12 investigates whether the following video packet contains the sequence header code (without it increments the video pointer vpp) at step S39. When the packet contains the sequence header code, processing progresses to step S47. When the packet does not contain the sequence header code, processing progresses to step S40.
[0083] The stream analysis section 12 increments the pointer vpp of a video packet at step S40.
[0084] The stream analysis section 12 is step S41, and investigates whether P picture or I picture was completed. When P picture or I picture is not completed, processing returns to step S39. When P picture or I picture is completed, processing progresses to step S42.
[0085] The stream analysis section 12 makes P1_end_address the address of the packet which is step S42 and P or I picture ends (drawing 8). It means that the address of P picture of the next beginning of I picture or I picture had been determined by the above.
[0086] The stream analysis section 12 investigates whether the following video packet contains the sequence header code (without it increments the video pointer vpp) at step S43. When the video packet contains the sequence header code, processing progresses to step S47. When the video packet does not contain the sequence header code, processing progresses to step S44.
[0087] The stream analysis section 12 increments the pointer vpp of a video packet at step S44.
[0088] The stream analysis section 12 investigates whether P picture or I picture was completed at step S45. When P picture or I picture is not completed, processing returns to step S43. When P picture or I picture is completed, processing progresses to step S46.
[0089] The stream analysis section 12 makes P2_end_address the address of the packet which is step S46 and P or I picture ends (drawing 8). It means that the address of the next next P picture of I picture or I picture had been determined by the above.
[0090] The stream analysis section 12 outputs the address of I_start_address, I_end_address, P1_end_address, and P2_end_address to the entry point map creation section 16 at step S47. In addition, at least one side of

P1_end_address and P2_end_address may not exist at this time.
[0091] The stream analysis section 12 is step S48, and judges whether a current packet is the last input packet. When a current packet is not the last packet, processing returns to step S33. Processing is ended when a current packet is the last packet.
[0092] Analysis of the above video stream is performed to the video packet of each program, when two or more programs are in the transport stream to record.
[0093] The stream analysis section 12 will supply this to the entry point map creation section 16, if entry point data are generated as mentioned above. The entry point map creation section 16 creates an entry point map as table-izes the entry point data supplied from the stream analysis section 12 for every program and shows them to drawing 7 , and supplies it to the file system section 17.
[0094] The block unit map and entry point map as focus data which express the focus as the transport stream by which the time stump was added to the file system section 17 by the time stump adjunct 15 as mentioned above are supplied, respectively from the block unit map creation section 23 and the entry point map creation section 16. The file system section 17 file-izes a transport stream and the focus data corresponding to it.
[0095] Drawing 24 expresses the example of this file structure. In this example, three programs are multiplexed in the transport stream file. As shown in this drawing, the entry point map is considered as the configuration subordinate to a block unit map. And each entry point map has the following data for every program, respectively. (1) program_number of a program (2) PID of the transport packet of PMT of a program (3) PID and stream_type (4) of a transport packet of video which constitute a program A program PID and stream_type (5) of a transport packet of an audio to constitute PID of PCR of a program (6) The file generated by the list file system section 17 of an entry point After the error correction section 18 is supplied and an error correcting code is added, the modulation section 19 is supplied and it becomes irregular by the predetermined method. The signal outputted from the modulation section 19 is supplied to the write-in section 20, and is written in a record medium 21.
[0096] A transport stream and its focus data are recorded on a record medium 21 as mentioned above.
[0097] Although the block unit map and the entry point map were created from the transport stream above, when the dynamic-image recording device itself multiplexes and generates a transport stream, a block unit map and an entry point map can be created at the time of the multiplexing actuation, for example. Drawing 25 expresses the example of a configuration in this case.
[0098] that is, in the example of drawing 25 , the video and the audio of the program of plurality (n pieces) are elementary in the multiplexing vessel 40 — stream #1 thru/or #n are inputted. The system time clock section 42 counts a system time clock with a frequency of 27MHz, generates a time stump, and is outputting it to a controller 41 and the block unit map creation section 43. A controller 41 analyzes each elementary stream inputted into the multiplexing machine 40, and the multiplexing machine 40 is controlled for the multiplexing machine 40 to fill T-STD (Transport Stream System Target Decoder) of MPEG 2 system specification, and to multiplex a transport stream.
[0099] A controller 41 outputs the packet number which is outputted from the multiplexing machine 40 and which shows the number of transport packets to the block unit map creation section 43 and the entry point map creation section 44. The block unit map creation section 43 generates a block unit map based on the time stump inputted as the packet number inputted from a controller 41 from the system time clock 42.
[0100] A controller 41 outputs program information and entry point data to the entry point map creation section 44 again. The entry point map creation section 44 generates an entry point map based on the packet number supplied from a controller 41, program information and entry point data, and the block unit map supplied to a list from the block unit map creation section 43.
[0101] The block unit map created by the transport stream and the block unit map creation section 43 which were outputted from the multiplexing machine 40, and the entry point map created by the entry point map creation section 44 are supplied to the file system section 17 shown in drawing 20 , respectively. The configuration to the file system section 17 thru/or a record medium 21 is the same as that of the case where it is shown in drawing 20 .
[0102] In the dynamic-image recording device 1 of a configuration as shown in this drawing 25 , from the elementary stream multiplexed with the multiplexing vessel 40, a controller 41 generates program information and entry point data, and outputs to the entry point map creation section 44. Moreover, a controller 41 outputs the packet number corresponding to the time stump inputted from the system time clock 42 to the block unit map creation section 43 and the entry point map creation section 44.
[0103] The block unit map creation section 43 creates a block unit map based on the time stump inputted as the packet number inputted from a controller 41 from the system time clock 42. Similarly, the entry point map creation section 44 creates an entry point map based on the packet number inputted from a controller 41, program information and entry point data, and the block unit map inputted into a list from the block unit map creation section 43.
[0104] And like the case where the transport stream, block unit map, and entry point map which were created are shown in drawing 20 , it is file-ized by the file system section 17, and a part for an error correction is added by the error correction section 18. And after the modulation section 19 becomes irregular further, it is recorded on a record medium 21 by the write-in section 20.
[0105] Next, the dynamic-image regenerative apparatus which reproduces the record medium 21 with which a transport stream file and the focus data of the stream were recorded as mentioned above is explained. Drawing 26 expresses the example of a configuration of such a dynamic-image regenerative apparatus 51. The read-out section 61 reads the data currently recorded on the record medium 21, and outputs them to the recovery section 62. The

recovery section 62 restores to the data inputted from the read-out section 61, and outputs them to the error correction section 63. The error correction section 63 corrects the error of the data inputted from the recovery section 62, and supplies it to the file system section 64.
[0106] The file system section 64 outputs focus data to the playback control section 67 while it divides into a transport stream file and focus data the data inputted from the error correction section 63 and supplies a file stream file to a demultiplexer 65. The playback control section 67 controls the read-out section 61, a demultiplexer 65, and the AV decoder 66 through a user interface corresponding to the command inputted by the ** user from a terminal 69.
[0107] From the transport stream file inputted from the file system section 64, a demultiplexer 65 extracts the video data and audio data of the channel corresponding to the command from the playback control section 67, and outputs them to the AV decoder 66. The AV decoder 66 decodes the video data and audio data which were inputted from the demultiplexer 65, and outputs them from a terminal 68.
[0108] Next, the actuation is explained. The transport stream file recorded with the dynamic-image recording apparatus 1 of drawing 20 (or drawing 25) and the focus data of the stream are recorded on the record medium 21. One or more programs are multiplexed by the transport stream file.
[0109] The playback control section 67 directs to read the focus data of a stream to the read-out section 61 first. At this time, the read-out section 61 reads the focus data of a stream from a record medium 21, and outputs them to the recovery section 62. The recovery section 62 restores to the inputted data, and outputs them to the error correction section 63. The error correction section 63 corrects the error of the inputted data, and supplies it to the file system section 64. The file system section 64 outputs the inputted stream focus data to the playback control section 67.
[0110] From a terminal 69, the program number which had playback specified is inputted by the user interface, and it is inputted into the playback control section 67. The playback control section 67 reads PID of PCR from focus data to PID of the transport packet of PMT of the program, PID of the transport packet of the video which constitutes a program, PID of the transport packet of the audio which constitutes stream_type and a program, stream_type, and a list, and outputs it to them to a demultiplexer 65 and the AV decoder 66.
[0111] Furthermore, the playback control section 67 directs to read a transport stream file to the read-out section 61. Corresponding to this command, the read-out section 61 reads a transport stream file from a record medium 21. This data is inputted into a demultiplexer 65 through processing of the recovery section 62, the error correction section 63, and the file system section 64 like the case where it mentions above.
[0112] A demultiplexer 65 separates the transport packet of video and an audio which constitutes the program specified by the user interface from the transport stream into which it was inputted, and inputs it into the AV decoder 66. The AV decoder 66 decodes a video stream and an audio stream, and outputs them from a terminal 68 as a playback video signal and a playback audio signal.
[0113] When random access playback is directed by the user interface, based on the contents of the focus data of the stream memorized inside, the playback control section 67 determines the read-out location of the data from a record medium 21, reads random access control information, and inputs it into the section 61. For example, when reproducing the program chosen by the user the middle from predetermined time of day, the playback control section 67 calculates the address of the transport stream corresponding to the specified time of day based on a block unit map, as it reads data from the address, it reads it, and it is directed to the section 61. Below, the procedure is explained.
[0114] Time-of-day block_unit_time (N) of the initial data of the Nth block unit is calculable as follows from the data of a block unit map.
[0115]
[Equation 1]

$$block\_unit\_time(N)=start\_time+\sum_{i=0}^{N-1} delta\_block\_unit\_time(i)$$

Here, delta_block_unit_time (i) is delta_block_unit_time of the i-th block unit. The block unit of i= 0 is BU0. And when N to which block_unit_time (N) becomes larger than the time of day specified by the user is known, it turns out that what is necessary is just to read data from the Nth block unit.
[0116] In this case, the address of the initial data of 0, then the Nth block unit (N> 0) is as follows about the address of the initial data of the 0th block unit on the transport stream currently recorded. When the data of the entry point map corresponding to first_block_unit_size+x(N-1) block_unit_size and the program chosen by the user exist, the playback control section 67 can control special playback based on entry point data. For example, in high-speed playback, as sequential continuation is carried out and the playback control section 67 reads the stream data of the address for every entry point, it is read, and it is directed to the section 61.
[0117] Drawing 27 expresses actuation of the playback control section 67 in this case. The playback control section 67 is step S61, and sets program_number of the program to reproduce to the memory to build in corresponding to the command from a user.
[0118] The playback control section 67 is step S62, and investigates from paesed_program_flag whether the entry point data of the program exist. When it exists (it is paesed_program_flag=1), it progresses to step S63. Since the data access which used the entry point map is not made when entry point data do not exist, processing is ended.
[0119] The playback control section 67 is step S63, and calculates the number BN of the block unit read and

started from the time of day specified by the user as mentioned above. That is, the number BN of the block unit to which the value (time of day of the head of a block unit) calculated with the above-mentioned [-one number] becomes larger than the specified time of day is calculated.

[0120] The playback control section 67 is step S64, and investigates from entry_point_flag whether the entry point of the program exists in a BN position block unit. When an entry point exists (it is entry_point_flag=1), it progresses to step S65, and when it does not exist, it progresses to step S67.

[0121] When an entry point exists, the playback control section 67 is step S65, and calculates the address which reads the stream data of an entry point from entry_point_data(). The read-out starting address of stream data is I_start_address, and a read-out ending address is I_end_address, P1_end_address, or P2_end_address.

[0122] The playback control section 67 is step S66, based on the address calculated at step S65, as it reads the stream data of an entry point, it is read, and it is directed in the section 61. The read-out section 61 performs read-out actuation corresponding to these directions.

[0123] The playback control section 67 is step S67, and increments a number BN. When it judges whether it was ordered in termination of processing and ordered in termination of processing, the playback control section 67 is step S68, ends return to step S64, and when that is not right, it ends processing.

[0124] The read-out section 61 reads data from the specified random access point. Through processing of the recovery section 62, the error correction section 63, and the file system section 64, the read data are inputted into a demultiplexer 65, and are decoded and outputted by the AV decoder 66.

[0125] The detail of the computation of this step S63 is further explained with reference to the flow chart of drawing 28 and drawing 29 . In step S81, if the playback start time Tst is inputted into the playback control section 67 as program_number from a terminal 69, in step S82, it will judge whether the playback control section 67 has the playback start time Tst equal to start time start_time (drawing 3 (B)) of the transport stream contained in focus data inputted at step S81. When the playback start time Tst is equal to start time start_time, it progresses to step S86, and the playback control section 67 sets 0 as the variable N showing the number of a block unit, and sets 0 as block_unit_address (N) of the block unit (0th block unit).

[0126] On the other hand, in step S82, when it judges that the playback start time Tst is not equal to start time starttime, it progresses to step S83, and the playback control section 67 reads the header unit of a block unit map, and calculates the minimum value N with which the following inequality is filled in step S84 based on a block unit map.

[0127] Tst<=block_unit_time (N)

Here, block_unit_time (N) is expressed with a degree type.

[0128]

[Equation 2]

$$\text{block\_unit\_time}(N)=\text{start\_time}+\sum_{i=0}^{N-1}\text{delta\_block\_unit\_time}(i)$$

In step S85, the playback control section 67 calculates address block_unit_address (N) of the Nth block unit from a degree type.

[0129]

block_unit_address (N) = first_block_unit_size When time-of-day block_unit_address (N) of the initial data of a +(N-1) xblock_unit_sizeN position block unit is calculated, in step S87, the playback control section 67 directs data read-out from address block_unit_address (N) of the Nth block unit in the read-out section 61.

[0130] The read-out section 61 reads the transport stream from address block_unit_address (N) from a record medium 21 in step S88 corresponding to the command from the playback control section 67. The read data are supplied to a demultiplexer 65 through the recovery section 62, the error correction section 63, and the file system section 64.

[0131] In step S89, the playback control section 67 outputs program_number of a program playback was instructed to be by the user to a demultiplexer 65. In step S90, a demultiplexer 65 separates the transport packet of the program of program_number directed from the playback control section 67, and outputs it to the AV decoder 66. In step S91, the AV decoder 66 decodes the data inputted from the demultiplexer 65, and outputs them from a terminal 68.

[0132] Next, the gestalt of the 2nd operation is explained. The dynamic-image recording device of the gestalt of this operation calculates the address on a break and the stream of the data for every time unit for the time amount on a stream to predetermined every time unit (unit time amount), when recording the transport stream by which one or more programs are multiplexed on record media, such as a disk and a tape. And the time unit map in which the address on the stream of the data for every time unit of this is shown is created. Furthermore, the entry point map in which the location of the entry point (random access point) for every program of a transport stream to record is shown is created. An entry point map has the structure subordinate to a time unit map. This time unit map is explained below.

[0133] Drawing 30 shows the transport stream by which two or more AV programs were multiplexed. An axis of abscissa shows time amount and is divided into every [of spacing of deltat] time unit TUi (i= 0, 1, 2 ...) here. The figure i which continues behind an alphabetic character TU shows the time order foreword of a time unit TU. The time amount length of all the time units TU is the same value deltat at the time of the first original record. Magnitude of value deltat is made into 0.5 seconds. It is chosen out of an input transport stream for record of one

or more AV programs. The selected transport packet attaches a slash and is shown. As the selected transport packet is generally shown in drawing 30 (B), it appears to irregular timing and the number of the transport packets for every time unit TUi of spacing of deltat changes.
[0134] As shown in drawing 31 , the selected transport packet packs spacing and is recorded on a record medium. At this time, the time stump in which the time of day on each stream is shown is added to each transport packet. Let this time stump be the same thing as TSP_extra_header of 4-byte length added to the transport packet specified by the DV format.
[0135] In drawing 31 , an axis of abscissa is the address which shows the byte position of the recorded transport stream. Moreover, the start address of the transport packet first inputted for every time unit on the axis of abscissa is shown. In this example, four pieces, three pieces, or six transport packets are recorded in time units TU0 and TU1 and TU2, respectively. The transport packet inputted ranging over two time units is included in the time unit by the side of before. The start address of a time unit TU0 and the transport packet inputted into the beginning of TU1 and TU2 shall be expressed as A (TU0), A (TU1), and A (TU2), respectively.
[0136] Drawing 32 shows the example of a time unit map, i.e., the table of the start address of the data for every time unit of the recorded transport stream. Here, time_unit_address shows the address of the initial data of the time unit on the recorded stream. On a time unit map, data length delta_time_unit_address for every time unit is table-ized.
[0137] In this example, the data length of a time unit TU0 is expressed with the difference (A(TU1)-A (TU01)) of the address A of the head of a time unit TU1 (TU1), and the address A of the head of a time unit TU0 (TU0). Similarly, the data length of a time unit TU1 is expressed with the difference (A(TU2)-A (TU1)) of the address A of the head of a time unit TU2 (TU2), and the address A of the head of a time unit TU1 (TU1), and the data length of a time unit TU2 is expressed in the difference (end_address-A (TU2)) of the address A of the head of a time unit TU2 (TU2) as address end_address of the last of a time unit TU2.
[0138] Next, an above-mentioned entry point map is explained. The transport streams shown in drawing 33 are the transport stream shown in drawing 31 , and same transport stream. The entry point shall have begun in the transport packet shown with a slash here. Specifically in an entry point, the sequence header and I picture data of MPEG video shall have begun. When an entry point exists in a predetermined time unit, the offset address from the start address of the data of the time unit to the address of an entry point is calculated. That is, in the example of drawing 33 , an entry point (I picture) exists in time units TU0 and TU2. Then, in a time unit TU0, the spacing a from the address A of the head (TU0) to address I_start_address of the head of I picture is calculated as an offset address. In a time unit TU2, the spacing b from the address A of the head (TU2) to address I_start_address of the head of I picture is calculated as an offset address by coincidence.
[0139] Drawing 34 shows the example of an entry point map, i.e., the table of the offset address to the entry point for every time unit. entry_point_flag is set to "1" when an entry point exists in the corresponding time unit TUi, and when it does not exist, it is set to "0." About the time unit whose entry_point_flag is "1", offset address I_start_offset_from_time_unit_address from start-address time_unit_address of the data of the time unit to address I_start_address of an entry point is calculated, as shown in a degree type.
[0140] I_start_offset_from_time_unit_address= I_start_address - For every time_unit_address and entry point, ending-address P2_end_address of the next next P of the next P of ending-address I_end_address of I picture data of an entry point and I picture of an entry point or ending-address P1_end_address of I picture, and I picture of an entry point or I picture is calculated, as shown in a degree type.
[0141] I_end_offset_address = I_end_address-I_start_addressP1_end_offset_address= P1_end_address-I_start_addressP2_end_offset_address= P2_end_address-I_start_address — the example of these addresses is shown in drawing 35 . Drawing 35 shows the MPEG video data which begins from the head of a predetermined time unit. Here, I, P, and B express I picture, P picture, or B picture, respectively, and the figure of a suffix shows the display order of a picture. I picture of the entry point shown by I2 exists in this time unit. Moreover, the next P picture of the I picture I2 is P5, and the next next P picture of the I picture I2 is P8. At this time, I_start_offset_from_time_unit_address calculated by the above-mentioned formula, I_end_offset_address, P1_end_offset_address, and P2_end_offset_address become the relation shown in drawing.
[0142] That is, let I_end_offset_address be the value which subtracted starting address I_start_address of the I picture I2 from ending-address I_end_address of the I picture I2. Let P1_end_offset_address be the value which subtracted starting address I_start_address of the I picture I2 from ending-address P1_end_address of the P picture P5. Furthermore, let P2_end_offset_address be the value which subtracted starting address I_start_address of the I picture I2 from ending-address P2_end_address of the P picture P8.
[0143] Let I_start_offset_from_time_unit_address be the value which subtracted address time_unit_address of the head of the data of a time unit from address Istart_address of an entry point.
[0144] In addition, when two or more programs are included in the transport stream to record, the information on an entry point is distinguished and created for every program. Moreover, it has the information (parsed_program_flag) which shows whether, as for an entry point map, entry point data exist for every program in consideration of the case where entry point data cannot be prepared, about all programs.
[0145] When the transport stream recorded on the record medium is edited, the time unit map is changed (updating). Next, the approach is explained. Drawing 36 (A) shows the example in the case of eliminating two packets of the head of a transport stream shown in drawing 31 , and the last three packets. Drawing 36 (B) shows the transport stream after doing in this way and carrying out partial elimination of the packet. Drawing 37 shows the time unit map

of the transport stream of drawing 36 (B). Thus, since the time amount length (first_time_unit_size) of the first time unit TU0 changes when the data to the middle of a time unit are eliminated, this is rewritten.
[0146] the case of drawing 36 (B) — the time amount length of a time unit TU0 — the difference of the time stump (A (TU1)) of the packet Pb of the head of a time unit TU1, and the time stump (C) of the packet Pa of the head of the time unit TU0 after elimination — it is changed into a value (A(TU1)-C). Moreover, the time amount length of a time unit TU2 is updated by the difference (D-A (TU2)) of the address D of the packet of the last of the time unit TU2 after elimination, and the address A of the packet of the head of the time unit TU2 (TU2). When a time unit map is changed, the entry point map related to it is also changed.
[0147] Next, the example of the syntax of an above-mentioned time unit map is shown in drawing 38 and drawing 39 . Drawing 38 and drawing 39 are the header unit (TimeUnitMapHeader()) of a time unit map, respectively. Data division (TimeUnitMapData()) It expresses. When recording a time unit map as a file, you may record by making a header unit and data division one file, and may record as a separate file. start_time of TimeUnitMapHeader(), and end_time The recording start time of day and record end time when the start time and end time of this time unit map being shown, for example, recording a certain transport stream, respectively are shown. first_time_unit_size shows the time amount length of the first time unit. time_unit_size The time amount length of the time unit of the 2nd henceforth is shown. number_of_time_unit_entries shows the number of the time units in a transport stream. A number of delta_time_unit_address (drawing 32) shown by number_of_time_unit_entries is written to TimeUnitMapData().
[0148] Moreover, the 1st example of the syntax of an above-mentioned entry point map is shown in drawing 40 and drawing 41 . In addition, header unit of an entry point map (EntryPointMapHeader()) Since it is the same as that of the case where it is shown in drawing 13 , a configuration is omitted here. Drawing 40 expresses the data division (EntryPointMapData()) of an entry point map. Drawing 41 expresses the syntax of entry_point_data() of drawing 40 further. When recording an entry point map as a file, you may record by making a header unit and data division one file, and may record as a separate file.
[0149] The data of the entry point about each program to record are described by EntryPointMapData() of drawing 40 . The parameters of the entry point about one time unit are entry_point_flag and entry_point_data(). The contents of entry_point_data() about one time unit are entry_point_time_stamp, I_start_offset_from_time_unit_address, I_end_offset_address, and P1_end_offset_address, as shown in drawing 41 . It is P2_end_offset_address. Here, entry_point_time_stamp is calculated based on PTS (Presentation Time Stamp) of the time of day on the stream of the transport packet of an entry point, or I picture of an entry point. PTS is PES of MPEG 2 systems specification. It is the information added to the header of a packet.
[0150] Moreover, since it is the same as that of the case where it is shown in drawing 16 , the 2nd example of the syntax of an above-mentioned entry point map is omitted here. The configuration of EntryPointMapHeader() and entry_point_data() is the same as that of the case where it is shown in the 1st above-mentioned drawing 13 or above-mentioned drawing 41 in an example. Drawing 16 is compared with drawing 40 , and the data list direction of the entry point about each program differs from the 1st example of drawing 40 so that clearly.
[0151] Next, the 1st example of the condition which shows below, and the example of the data list of the entry map in the case of [each] the 2nd example are shown. Here, as shown in drawing 42 , three programs (program#1, program#2, program#3) shall be multiplexed in the transport stream, and every time unit TUi (i= 0, 1, 2, 3) shall have the entry point of each program. In this case, each parameter is as follows.
[0152]
number_of_time_unit_entries =4 number_of_programs=3 program_number=1:parsed_program_flag=1 program_number=2:parsed_program_flag=1 program_number=3:parsed_program_flag=1 NUMBER_OF_ParsedPrograms=3
drawing 43 shows the entry point map in the case of the 1st example (example of drawing 40). In this case, it becomes the form where the list of entry point data separated for every program. That is, since entry_point_data#1-1 thru/or entry_point_data#1-4 exist in a time unit TU0 thru/or each of TU3 as entry_point_data as EntryPointMapData of program#1 is shown in drawing 43 (A), entry_point_flag is set to "1", respectively.
[0153] In addition, entry_point_data#A-B expresses entry_point_data() about the Bth entry point of program_number=A.
[0154] As EntryPointMapData of program#2 is shown in drawing 43 (B), since entry_point_data does not exist, the entry_point_flag is set to "0" at a time unit TU1 and TU3. On the other hand, in a time unit TU0 and TU2, since entry_point_data#2-1 and entry_point_data#2-2 exist, respectively, the entry_point_flag is set to "1."
[0155] Furthermore, since entry_point_data does not exist, entry_point_flag is set to "0" at the time unit TU0 of EntryPointMapData of program#3, and TU2. Since entry_point_data#3-1 and entry_point_data#3-2 exist, respectively, the entry_point_flag is set to "1" at a time unit TU1 and TU3.
[0156] entry_point_data is described to be these entry_point_flag by EntryPointMapData.
[0157] Moreover, drawing 44 shows the entry point map in the case of the 2nd example (example of drawing 16).
[0158] In this case, it becomes the form where the entry point data of each program are located in a line with time order for every time unit, and the list of entry point data becomes one form. That is, in a time unit TU0, three program program#1 thru/or #3 are described and entry_point_flag and corresponding entry_point_data are described about each. In this example, since that entry_point_flag is set to "0" since entry_point_data does not exist, and entry_point_data#1-1 and #2-1 exist in program#3 about program#1 and #2, that entry_point_flag is set to "1."
[0159] the other time units TU1 thru/or TU3 — also setting — program#1 thru/or #3 — entry_point_flag and entry_point_data are described about each.

[0160] Next, the example of a configuration of the dynamic-image recording device 1 which creates an above-mentioned table and is recorded on a record medium with a transport stream from the inputted transport stream is shown in drawing 45 .
[0161] In this example of a configuration, the block unit map creation section 23 in drawing 20 is changed into the time unit map creation section 123. Since other configurations and actuation are the same as that of the case in drawing 20 thru/or drawing 23 , the detailed explanation is omitted.
[0162] The stream analysis section 12 will supply this to the entry point map creation section 16, if entry point data are generated. The entry point map creation section 16 creates an entry point map as table-izes the entry point data supplied from the stream analysis section 12 for every program and shows them to drawing 34 , and supplies it to the file system section 17.
[0163] The time unit map and entry point map as focus data which express the focus as the transport stream to which the time stump was added by the time stump adjunct 15 are supplied to the file system section 17, respectively from the time unit map creation section 123 and the entry point map creation section 16. The file system section 17 file-izes a transport stream and the focus data corresponding to it.
[0164] Drawing 46 expresses the example of this file structure. In this example, three programs are multiplexed in the transport stream file. As shown in this drawing, the entry point map is considered as the configuration subordinate to a time unit map. Each entry point map of the data which it has for every program is the same as that of the case in drawing 24 .
[0165] After the error correction section 18 is supplied and an error correcting code is added, the file generated by the file system section 17 is supplied to the modulation section 19, and is modulated by the predetermined method. The signal outputted from the modulation section 19 is supplied to the write-in section 20, and is written in a record medium 21.
[0166] A transport stream and its focus data are recorded on a record medium 21 as mentioned above.
[0167] Although the time unit map and the entry point map were created from the transport stream above, when the dynamic-image recording device itself multiplexes and generates a transport stream, a time unit map and an entry point map can be created at the time of the multiplexing actuation, for example. Drawing 47 expresses the example of a configuration in this case.
[0168] In the example of drawing 47 , the block unit map creation section 43 in the example of drawing 25 is changed into the time unit map creation section 143. Since it is the same as that of the case in drawing 25 , other configurations and actuation are omitted here.
[0169] The dynamic-image regenerative apparatus which reproduces the record medium 21 with which a transport stream file and the focus data of the stream were recorded as mentioned above becomes being the same as that of the case where it is shown in drawing 26 .
[0170] Next, the actuation is explained. The transport stream file recorded with the dynamic-image recording apparatus 1 of drawing 45 (or drawing 47) and the focus data of the stream are recorded on the record medium 21. One or more programs are multiplexed by the transport stream file.
[0171] The playback control section 67 directs to read the focus data of a stream to the read-out section 61 first. At this time, the read-out section 61 reads the focus data of a stream from a record medium 21, and outputs them to the recovery section 62. The recovery section 62 restores to the inputted data, and outputs them to the error correction section 63. The error correction section 63 corrects the error of the inputted data, and supplies it to the file system section 64. The file system section 64 outputs the inputted stream focus data to the playback control section 67.
[0172] From a terminal 69, the program number which had playback specified is inputted by the user interface, and it is inputted into the playback control section 67. The playback control section 67 reads PID of PCR from focus data to PID of the transport packet of PMT of the program, PID of the transport packet of the video which constitutes a program, PID of the transport packet of the audio which constitutes stream_type and a program, stream_type, and a list, and outputs it to them to a demultiplexer 65 and the AV decoder 66.
[0173] Furthermore, the playback control section 67 directs to read a transport stream file to the read-out section 61. Corresponding to this command, the read-out section 61 reads a transport stream file from a record medium 21. This data is inputted into a demultiplexer 65 through processing of the recovery section 62, the error correction section 63, and the file system section 64 like the case where it mentions above.
[0174] A demultiplexer 65 separates the transport packet of video and an audio which constitutes the program specified by the user interface from the transport stream into which it was inputted, and inputs it into the AV decoder 66. The AV decoder 66 decodes a video stream and an audio stream, and outputs them from a terminal 68 as a playback video signal and a playback audio signal.
[0175] When random access playback is directed by the user interface, based on the contents of the focus data of the stream memorized inside, the playback control section 67 determines the read-out location of the data from a record medium 21, reads random access control information, and inputs it into the section 61. For example, when reproducing the program chosen by the user the middle from predetermined time of day, the playback control section 67 calculates the address of the transport stream corresponding to the specified time of day based on a time unit map, as it reads data from the address, it reads it, and it is directed to the section 61. Below, the procedure is explained.
[0176] The time of day of start_time, then the initial data of the Nth time unit (N> 0) serves as (start_time+first_time_unit_size+(N-1) * time_unit_size) in the time of day of the initial data of the time unit TU0 of

eye zero watch. When the number of the time unit to which the time of day of the initial data of a time unit becomes large rather than the time of day specified by the user is known, it turns out that what is necessary is just to read data from the time unit of the number.
[0177] In this case, 0, then address time_unit_address (N) of the initial data of the Nth time unit can calculate the address of the initial data of the 0th time unit on the recorded stream as follows.
[0178]
[Equation 3]

$$time\_unit\_address(N)=\sum_{i=0}^{N-1} delta\_time\_unit\_address(i)$$

Moreover, when the data of the entry point map corresponding to the program chosen by the user exist, the playback control section 67 can control special playback based on entry point data. For example, in high-speed playback, as sequential continuation is carried out and the playback control section 67 reads the stream data of the address for every entry point, it is read, and it is directed to the section 61.
[0179] Drawing 48 expresses actuation of the playback control section 67 in this case. The playback control section 67 is step S161, and sets program_number of the program to reproduce to the memory to build in corresponding to the command from a user.
[0180] The playback control section 67 is step S162, and investigates from paesed_program_flag whether the entry point data of the program exist. When it exists (it is paesed_program_flag=1), it progresses to step S163. Since the data access which used the entry point map is not made when entry point data do not exist, processing is ended.
[0181] The playback control section 67 is step S163, and calculates the number TN of the time unit read and started from the time of day specified by the user as mentioned above. That is, the number TN of the time unit to which the value (time of day of the head of a time unit) of start_time+first_time_unit_size+(N-1) *time_unit_size becomes larger than the specified time of day is calculated.
[0182] The playback control section 67 is step S164, and investigates from entry_point_flag whether the entry point of the program exists in a TN position time unit. When an entry point exists (it is entry_point_flag=1), it progresses to step S165, and when it does not exist, it progresses to step S167.
[0183] When an entry point exists, the playback control section 67 is step S165, and calculates the address which reads the stream data of an entry point from entry_point_data(). The read-out starting address of stream data is I_start_address, and a read-out ending address is I_end_address, P1_end_address, or P2_end_address.
[0184] The playback control section 67 is step S166, based on the address calculated at step S165, as it reads the stream data of an entry point, it is read, and it is directed in the section 61. The read-out section 61 performs read-out actuation corresponding to these directions.
[0185] The playback control section 67 is step S167, and increments a number TN. When it judges whether it was ordered in termination of processing and ordered in termination of processing, the playback control section 67 is step S168, ends return to step S164, and when that is not right, it ends processing.
[0186] The read-out section 61 reads data from the specified random access point. Through processing of the recovery section 62, the error correction section 63, and the file system section 64, the read data are inputted into a demultiplexer 65, and are decoded and outputted by the AV decoder 66.
[0187] The detail of the computation of this step S163 is further explained with reference to the flow chart of drawing 49 and drawing 50 . In step S181, if the playback start time Tst is inputted into the playback control section 67 as program_number from a terminal 69, in step S182, it will judge whether the playback control section 67 has the playback start time Tst equal to start time start_time (drawing 30 (B)) of the transport stream contained in focus data inputted at step S181. When the playback start time Tst is equal to start time start_time, it progresses to step S186, and the playback control section 67 sets 0 as the variable N showing the number of a time unit, and sets 0 as time_unit_address (N) of the time unit (0th time unit).
[0188] On the other hand, in step S182, when it judges that the playback start time Tst is not equal to start time starttime, it progresses to step S183, and the playback control section 67 reads the header unit of a time unit map, and calculates the minimum value N with which the following inequality is filled in step S184.
[0189] In the Tst<=start_time+first_time_unit_size+(N-1) xtime_unit_size step S185, the playback control section 67 calculates time_unit_address (N) according to the formula shown by several 1 based on the data of a time unit map.
[0190] When time-of-day time_unit_address (N) of the initial data of the Nth time unit is calculated, in step S187, the playback control section 67 directs data read-out from address time_unit_address (N) of the Nth time unit in the read-out section 61.
[0191] The read-out section 61 reads the transport stream from address time_unit_address (N) from a record medium 21 in step S188 corresponding to the command from the playback control section 67. The read data are supplied to a demultiplexer 65 through the recovery section 62, the error correction section 63, and the file system section 64.
[0192] In step S189, the playback control section 67 outputs program_number of a program playback was instructed to be by the user to a demultiplexer 65. In step S190, a demultiplexer 65 separates the transport packet of the program of program_number directed from the playback control section 67, and outputs it to the AV decoder 66. In step S191, the AV decoder 66 decodes the data inputted from the demultiplexer 65, and outputs them from a terminal 68.
[0193] Although a series of processings mentioned above can also be performed by hardware, they can also be

performed with software. When performing a series of processings with software, the program which constitutes the software is installed in a general-purpose personal computer etc. possible [performing various kinds of functions] by installing the computer built into the dynamic-image record regenerative apparatus as hardware of dedication, or various kinds of programs.

[0194] Next, the case where the computer is a general-purpose personal computer is explained as an example about the medium used in order to install in a computer the program which performs a series of processings mentioned above and to make it into the condition which can be performed by computer with reference to drawing 51 .

[0195] As shown in drawing 51 (A), a user can be provided with a program in the condition of having installed in the hard disk 302 and semiconductor memory 303 as a record medium which are built in the computer 301 beforehand.

[0196] Or as shown in drawing 51 (B), a program can be stored in record media, such as a floppy disk 311, CD-ROM (Compact Disk-Read Only Disk)312, the MO (Magneto-Optical) disk 313, DVD (Digital Versatile Disk)314, a magnetic disk 315, and semiconductor memory 316, temporarily or permanently, and can be offered as a software package again.

[0197] Furthermore, it can transmit to a computer 323 on radio, or a program can be transmitted to a computer 323 with a cable through a Local Area Network and a network 131 called the Internet, and can be made to store in the hard disk to build in in a computer 323 through the satellite 322 for digital satellite broadcasting from the download site 321, as shown in drawing 51 (C).

[0198] The medium in this specification means the concept of the wide sense containing all these media.

[0199] Moreover, in this specification, even if the processing serially performed in accordance with the sequence that the step which describes the program offered by the medium was indicated is not of course necessarily processed serially, it is a juxtaposition thing also including the processing performed according to an individual.

[0200] In addition, in this specification, a system expresses the whole equipment constituted by two or more equipments.

[0201] Thus, since the starting position of I picture or an audio frame can be efficiently searched when carrying out random access to the record medium with which one or more transport streams were recorded and reproducing, quick random access playback of a response can be carried out to a user input.

[0202]

[Effect of the Invention] Since the unit map including the amount-of-data information which carries out unitization of the coding stream and expresses the amount of the data for every unit was created like the above according to the data processor according to claim 1, the data-processing approach according to claim 8, and the medium according to claim 9, the quick random access of a response becomes possible.

[Translation done.]

* NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]
[Drawing 1] It is drawing explaining the packet of the conventional transport stream.
[Drawing 2] It is drawing explaining the transport stream on which the former is recorded.
[Drawing 3] It is drawing explaining the transport stream of this invention.
[Drawing 4] It is drawing explaining the transport stream on which this invention is recorded.
[Drawing 5] It is drawing showing the example of a block unit map.
[Drawing 6] It is drawing explaining the offset address for every block unit.
[Drawing 7] It is drawing showing the example of an entry point map.
[Drawing 8] It is drawing explaining entry point data.
[Drawing 9] It is drawing explaining elimination of data.
[Drawing 10] It is drawing showing the example of the block unit map when eliminating data.
[Drawing 11] It is drawing showing the syntax of BlockUnitMapHeader().
[Drawing 12] It is drawing showing the syntax of BlockUnitMapData().
[Drawing 13] It is drawing showing the syntax of EntryPointMapHeader().
[Drawing 14] It is drawing showing the syntax of EntryPointMapData().
[Drawing 15] entry point It is drawing showing the syntax of data().
[Drawing 16] It is drawing showing the syntax of EntryPointMapData().
[Drawing 17] It is drawing explaining the entry point of a transport stream file.
[Drawing 18] It is drawing showing the example of EntryPointMapData.
[Drawing 19] It is drawing showing the example of EntryPointMapData.
[Drawing 20] It is the block diagram showing the example of a configuration of the dynamic-image recording apparatus which applied this invention.
[Drawing 21] It is a flow chart explaining actuation of the dynamic-image recording apparatus of drawing 20 .
[Drawing 22] It is a flow chart explaining actuation of the dynamic-image recording apparatus of drawing 20 .
[Drawing 23] It is a flow chart explaining actuation of the dynamic-image recording apparatus of drawing 20 .
[Drawing 24] It is drawing explaining the relation between the block unit map of a transport stream file, and an entry point map.
[Drawing 25] It is the block diagram showing other examples of a configuration of the dynamic-image recording apparatus which applied this invention.
[Drawing 26] It is the block diagram showing the example of a configuration of the dynamic-image regenerative apparatus which applied this invention.
[Drawing 27] It is a flow chart explaining actuation of the dynamic-image regenerative apparatus of drawing 26 .
[Drawing 28] It is a flow chart explaining actuation of the dynamic-image regenerative apparatus of drawing 26 .
[Drawing 29] It is a flow chart explaining actuation of the dynamic-image regenerative apparatus of drawing 26 .
[Drawing 30] It is drawing explaining the transport stream of this invention.
[Drawing 31] It is drawing explaining the transport stream on which this invention is recorded.
[Drawing 32] It is drawing showing the example of a time unit map.
[Drawing 33] It is drawing explaining the offset address for every time unit.
[Drawing 34] It is drawing showing the example of an entry point map.
[Drawing 35] It is drawing explaining entry point data.
[Drawing 36] It is drawing explaining elimination of data.
[Drawing 37] It is drawing showing the example of the time unit map when eliminating data.
[Drawing 38] It is drawing showing the syntax of TimeUnitMapHeader().
[Drawing 39] It is drawing showing the syntax of TimeUnitMapData().
[Drawing 40] It is drawing showing the syntax of EntryPointMapData().
[Drawing 41] entry point It is drawing showing the syntax of data().
[Drawing 42] It is drawing explaining the entry point of a transport stream file.
[Drawing 43] It is drawing showing the example of EntryPointMapData.
[Drawing 44] It is drawing showing the example of EntryPointMapData.
[Drawing 45] It is the block diagram showing the example of a configuration of the dynamic-image recording apparatus which applied this invention.
[Drawing 46] It is drawing explaining the relation between the time unit map of a transport stream file, and an entry

point map.
[Drawing 47] It is the block diagram showing other examples of a configuration of the dynamic-image recording apparatus which applied this invention.
[Drawing 48] It is a flow chart explaining actuation of the dynamic-image regenerative apparatus of drawing 26 .
[Drawing 49] It is a flow chart explaining actuation of the dynamic-image regenerative apparatus of drawing 26 .
[Drawing 50] It is a flow chart explaining actuation of the dynamic-image regenerative apparatus of drawing 26 .
[Drawing 51] It is drawing explaining a medium.
[Description of Notations]
1 Dynamic-Image Recording Device 11 PID Filter, 12 The stream analysis section, 14 Time stump generating section 15 Time stump adjunct 16 Entry point map creation section 17 File system section 21 A record medium, 23 The time unit map creation section, 24 Counter 40 A multiplexing machine and 41 controller 42 System time clock section 43 Block unit map creation section 44 entry-point map creation section 61 Read-out section 65 Demultiplexer 66 AV decoder 67 playback control section

---

[Translation done.]

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2000－333128
（P2000－333128A）
(43)公開日　平成12年11月30日(2000.11.30)

(51)Int.Cl.'		識別記号	Ｆ Ｉ		テーマコード'（参考）	
Ｈ０４Ｎ	5/92		Ｈ０４Ｎ	5/92	Ｈ	５Ｃ０５３
	7/24			7/13	Ｚ	５Ｃ０５９

審査請求　未請求　請求項の数９　ＯＬ　（全 30 頁）

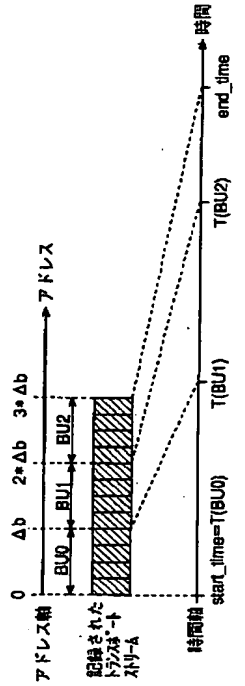
(21)出願番号　特願平11－137203
(22)出願日　平成11年５月18日(1999.5.18)
(31)優先権主張番号　特願平11－72303
(32)優先日　平成11年３月17日(1999.3.17)
(33)優先権主張国　日本（ＪＰ）

(71)出願人　000002185
ソニー株式会社
東京都品川区北品川６丁目７番35号
(72)発明者　加藤　元樹
東京都品川区北品川６丁目７番35号　ソニー株式会社内
(72)発明者　浜田　俊也
東京都品川区北品川６丁目７番35号　ソニー株式会社内
(74)代理人　100082131
弁理士　稲本　義雄

最終頁に続く

(54)【発明の名称】　データ処理装置および方法、並びに媒体

(57)【要約】
【課題】　記録媒体に対して、符号化ストリームを効率的に記録し、迅速にランダムアクセスできるようにする。
【解決手段】　入力されたトランスポートストリームから、記録が指定されたチャネルの所定の数のパケットが集められて、ブロックユニット化され、ダミーパケットが付加されることなく、記録媒体に記録される。各ブロックユニットBU0，BU1，BU2のパケットの内、記録されたパケットの先頭のアドレスblock_unit_addressと、例えば、タイムユニットBU1の先頭の時刻Ｔ(BU1)から、タイムユニットBU0の先頭の時刻Ｔ(BU0)を減算した値で表わされるブロックユニットBU0のデータ長が、対応され、ブロックユニットマップとして記録媒体に記録される。



【特許請求の範囲】
【請求項１】　入力された符号化ストリームに含まれるデータを処理するデータ処理装置において、
入力された前記符号化ストリームを、所定の単位のユニットにユニット化するユニット化手段と、
前記ユニット化手段によりユニット化された符号化ストリームを記憶する記憶手段と、
前記ユニット化手段によりユニット化された前記符号化ストリームのユニット毎のデータの量を表すデータ量情報を含むユニットマップを作成する第１の作成手段と、
前記符号化ストリームのプログラム毎のエントリポイントの位置を示す、前記ユニットマップに従属するエントリポイントマップを作成する第２の作成手段を備えることを特徴とするデータ処理装置。
【請求項２】　前記ユニットマップの前記データ量情報は、前記記憶手段のアドレスで表されていることを特徴とする請求項１に記載のデータ処理装置。
【請求項３】　前記ユニットマップの前記データ量情報は、前記記憶手段に記憶されている前記ユニットのデータ量に対応する時間で表されていることを特徴とする請求項１に記載のデータ処理装置。
【請求項４】　前記第２の作成手段は、前記符号化ストリームが編集されたとき、前記エントリポイントマップを変更することを特徴とする請求項１に記載のデータ処理装置。
【請求項５】　前記符号化ストリームとともに、前記ユニットマップまたはエントリポイントマップの少なくとも一方をファイル化するファイル化手段をさらに備えることを特徴とする請求項１に記載のデータ処理装置。
【請求項６】　前記ファイル化手段によりファイル化されたデータを記録媒体に記録する記録手段をさらに備えることを特徴とする請求項５に記載のデータ処理装置。
【請求項７】　前記第１の作成手段は、前記符号化ストリームが編集されたとき、前記ユニットマップを変更することを特徴とする請求項１に記載のデータ処理装置。
【請求項８】　入力された符号化ストリームに含まれるデータを処理するデータ処理装置のデータ処理方法において、
入力された前記符号化ストリームを、所定の単位のユニットにユニット化するユニット化ステップと、
前記ユニット化ステップの処理によりユニット化された符号化ストリームを記憶する記憶ステップと、
前記ユニット化ステップの処理によりユニット化された前記符号化ストリームのユニット毎のデータの量を表すデータ量情報を含むユニットマップを作成する作成ステップと、
前記符号化ストリームのプログラム毎のエントリポイントの位置を示す、前記ユニットマップに従属するエントリポイントマップを作成する第２の作成ステップを含むことを特徴とするデータ処理方法。
【請求項９】　入力された符号化ストリームに含まれるデータを処理するプログラムであって、
入力された前記符号化ストリームを、所定の単位のユニットにユニット化するユニット化ステップと、
前記ユニット化ステップの処理によりユニット化された符号化ストリームを記憶する記憶ステップと、
前記ユニット化ステップの処理によりユニット化された前記符号化ストリームのユニット毎のデータの量を表すデータ量情報を含むユニットマップを作成する作成ステップと、
前記符号化ストリームのプログラム毎のエントリポイントの位置を示す、前記ユニットマップに従属するエントリポイントマップを作成する第２の作成ステップを含むことを特徴とするプログラムをコンピュータに実行させる媒体。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、データ処理装置および方法、並びに媒体に関し、特に符号化ストリームから特徴点情報を抽出することにより、複数のプログラムが多重化されている場合においても、迅速にランダムアクセスができるようにしたデータ処理装置および方法、並びに媒体に関する。
【０００２】
【従来の技術】ヨーロッパのDVB(Digital Video Broadcast)や日本のディジタルBS放送などの多チャネルディジタルテレビジョン放送では、MPEG(Moving Picture Experts Group)２トランスポートストリームが使われる。トランスポートストリームは、トランスポートパケットが連続したストリームであり、トランスポートパケットは、例えば、MPEG２ビデオストリームやMPEG１オーディオストリームがパケット化されたものである。放送の電波で伝送される１本のトランスポートストリームには、１つまたは複数のAV(Audio Visual)プログラムが多重化されている。一般に、各チャネルのAVプログラムは、お互いに独立している。
【０００３】したがって、放送で送られるトランスポートストリームを家庭の受信機でそのまま受信し、記録すれば、そのトランスポートストリームのすべてのチャネルのプログラムを同時に記録することができる。また、放送で送られるトランスポートストリームの中からユーザによって選択された幾つかのチャネルのAVプログラムのトランスポートストリームを分離したものを記録すれば、選択された任意の数のチャネルのプログラムを同時に記録することができる。
【０００４】図１に従来のトランスポートストリームの記録方法の例を示す。図１(A)は、複数のAVプログラムが多重化されたトランスポートストリームを示す。ここで横軸は、時間であり、Δtの間隔のブロックユニットBUi(ｉ＝０，１，２・・・）毎に区切られている。入力

トランスポートストリームの中から１つまたは複数のAVプログラムが選択された時に、選択されたトランスポートパケットは斜線を施して示されている。選択されたトランスポートパケットは、一般に、図１(B)に示すように、不規則なタイミングで現れ、ブロックユニットBUi毎のトランスポートパケットの数は変化する。

【０００５】Δtの間隔のブロックユニットBUi毎の選択されたトランスポートパケットは、図２に示すように、間隔を詰めて記録媒体に記録される。この時、各トランスポートパケットは、それぞれのストリーム上の時刻を示すタイムスタンプを付加して記録される。このタイムスタンプは、例えば、DV(Digital Video) フォーマットで規定されているところのトランスポートパケットに付加される４バイト長のTSP_extra_headerと同様のものである。

【０００６】図２において、横軸は記録されたトランスポートストリームのバイト位置を示すアドレスである。図１(B)に示すような可変ビットレートのトランスポートストリームが入力されると、記録装置は、図２に示すように、ダミーデータを入れて、固定の記録レートでデータを記録する。したがって、記録されたトランスポートストリームの時間の経過に対するファイルのデータ量は比例する。すなわち、ブロックユニットあたりの記録データ量をxとすると、n番目（ｎ＝０，１，２，・・・）のブロックユニットの先頭データのバイト位置は、n倍のxとなる。

【０００７】

【発明が解決しようとする課題】このように、従来の記録方法は、ダミーデータを挿入して一定の記録レートにしているため、トランスポートストリームの記録効率が良くない。しかしながら、ダミーデータを挿入しないと、記録されたトランスポートストリームの時間の経過とファイルのデータ量が比例しなくなるので、トランスポートストリームの時間軸上の所定の位置のデータにアクセスする場合、データのアクセス性が悪くなる問題が発生する。

【０００８】また、一般に、MPEG２ビデオのストリームでは、０．５秒くらいの間隔でIピクチャが符号化され、それ以外のピクチャはPピクチャまたはBピクチャとして符号化される。したがって、MPEG２ビデオのストリームが記録された記録媒体から、ビデオ信号を高速再生する場合は、Iピクチャをサーチしなければならない。ところが、ディジタル放送等のトランスポートストリームが記録された記録媒体から、ランダムアクセスにより再生を行う場合、Iピクチャの開始バイトを効率よくサーチすることが難しかった。すなわち、記録媒体上のトランスポートストリームのランダムなバイト位置から読み出したビデオストリームのシンタックスを解析し、Iピクチャやオーディオフレームの開始バイトがサーチされていた。そのため、場合によっては、Iピクチャのサーチに時間がかかってしまい、ユーザ入力に対して応答の速いランダムアクセス再生をすることが困難であった。

【０００９】本発明は、このような状況に鑑みてなされたものであり、複数のプログラムが多重化されている場合においても、迅速に、ランダムアクセスができるようにするものである。

【００１０】

【課題を解決するための手段】請求項１に記載のデータ処理装置は、入力された符号化ストリームに含まれるデータを処理するデータ処理装置において、入力された符号化ストリームを、所定の単位のユニットにユニット化するユニット化手段と、ユニット化手段によりユニット化された符号化ストリームを記憶する記憶手段と、ユニット化手段によりユニット化された符号化ストリームのユニット毎のデータの量を表すデータ量情報を含むユニットマップを作成する第１の作成手段と、符号化ストリームのプログラム毎のエントリポイントの位置を示す、ユニットマップに従属するエントリポイントマップを作成する第２の作成手段を備えることを特徴とする。

【００１１】前記ユニットマップのデータ量情報は、記憶手段のアドレス、または記憶手段に記憶されているユニットのデータ量に対応する時間で表すことができる。

【００１２】前記第２の作成手段により、符号化ストリームが編集されたとき、エントリポイントマップを変更するようにすることができる。

【００１３】前記ファイル化手段によりファイル化されたデータを記録媒体に記録する記録手段をさらに設けることができる。

【００１４】前記第１の作成手段により、符号化ストリームが編集されたとき、ユニットマップを変更することができる。

【００１５】請求項８に記載のデータ処理方法は、入力された符号化ストリームに含まれるデータを処理するデータ処理装置のデータ処理方法において、入力された符号化ストリームを、所定の単位のユニットにユニット化するユニット化ステップと、ユニット化ステップの処理によりユニット化された符号化ストリームを記憶する記憶ステップと、ユニット化ステップの処理によりユニット化された符号化ストリームのユニット毎のデータの量を表すデータ量情報を含むユニットマップを作成する作成ステップと、符号化ストリームのプログラム毎のエントリポイントの位置を示す、ユニットマップに従属するエントリポイントマップを作成する第２の作成ステップを含むことを特徴とする。

【００１６】請求項９に記載の媒体のプログラムは、入力された符号化ストリームに含まれるデータを処理するプログラムであって、入力された符号化ストリームを、所定の単位のユニットにユニット化するユニット化ステップと、ユニット化ステップの処理によりユニット化された符号化ストリームを記憶する記憶ステップと、ユニ

ット化ステップの処理によりユニット化された　符号化ストリームのユニット毎のデータの量を表すデータ量情報を含むユニットマップを作成する作成ステップと、符号化ストリームのプログラム毎のエントリポイントの位置を示す、ユニットマップに従属するエントリポイントマップを作成する第２の作成ステップを含むことを特徴とする。

【００１７】請求項１に記載のデータ処理装置、請求項８に記載のデータ処理方法、および請求項９に記載の媒体においては、符号化ストリームがユニット化され、ユニット毎のデータの量を表すデータ量情報を含むユニットマップが作成される。

【００１８】

【発明の実施の形態】以下、符号化ストリームが１つまたは複数のプログラムが多重化されている多重化ストリームである場合を例として本発明を説明するが、本発明は、符号化ストリームがMPEGビデオストリーム等のエレメンタリーストリームであっても適用できるものである。

【００１９】最初に、本発明の基本的な原理について説明する。本発明の動画像記録装置は、１つまたは複数のプログラムが多重化されているトランスポートストリームをディスク、テープなどの記録媒体に記録する時に、所定のデータ量のブロックユニット毎に区切り、ブロックユニット毎のデータのストリーム上の時刻を計算する。そして、このブロックユニット毎のデータのストリーム上の時刻を示すブロックユニットマップが作成される。さらに、記録するトランスポートストリームのプログラム毎のエントリーポイント（ランダムアクセスポイント）の場所を示すエントリーポイントマップが作成される。エントリーポイントマップは、ブロックユニットマップに従属する構造を有する。このブロックユニットマップについて、以下に説明する。

【００２０】図３は、複数のAVプログラムが多重化されたトランスポートストリームを示す。ここで横軸は、ストリーム上の時刻を示す時間軸である。入力トランスポートストリームの中から１つまたは複数のAVプログラムが記録のために選択される。選択されたトランスポートパケットは斜線を付して示されている。選択されたトランスポートパケットは、一般に図３(B)に示すように、不規則なタイミングで現れる。

【００２１】選択されたトランスポートパケットは、図４に示すように、間隔を詰めて記録媒体に記録される。この時、各トランスポートパケットには、それぞれのストリーム上の時刻を示すタイムスタンプが付加される。このタイムスタンプは、例えば、DVフォーマットで規定されているところのトランスポートパケットに付加される４バイト長のTSP_extra_headerと同様のものとされる。

【００２２】図４には、横軸に、記録されたトランスポートストリームのバイト位置を示すアドレスと、ストリーム上の時刻が表されている。記録されたトランスポートストリームは、所定のデータ量Δbのブロックユニット毎に区切られる。この例では、Δbを4個のトランスポートパケット分のデータ量としている。ブロックユニットをBUと示す。BUの後ろに続く数字は、BUの時間順序を示す。最初のオリジナル記録の時は、すべてのBUのデータ長は同じくΔｂである。なお、Δbのデータ長は、この例に限らず、例えば、ECCブロックサイズの整数倍としても良い。

【００２３】図４の時間軸は、トランスポートストリーム上の時刻を示す時間軸を示す。この軸上に、ブロックユニット毎の先頭データの時刻を示す。ブロックユニットBU0，BU1，BU2の先頭データの時刻は、それぞれT(BU0)，T(BU1)，T(BU2)である。

【００２４】図５は、ブロックユニットマップ、すなわち記録されたトランスポートストリームのブロックユニット毎の先頭データのストリーム上の時刻のテーブルを示す。ここで、block_unit_addressは、記録されたストリーム上のブロックユニットの先頭データのアドレスである。また、block_unit_timeは、ブロックユニットの先頭データの時刻を表し、delta_block_unit_timeはブロックユニット毎の時間長を表す。ブロックユニットマップでは、ブロックユニット毎のdelta_block_unit_timeがテーブル化される。

【００２５】図５の例においては、ブロックユニットBU０の先頭のデータのアドレスblock_unit_addressは０とされ、対応する時刻block_unit_timeは、Ｔ（BU０）とされている。同様に、ブロックユニットBU１の先頭のデータアドレスblock_unit_addressは、Δｂとされ、対応する時刻block_unit_timeは、Ｔ（BU１）とされている。さらに、ブロックユニットBU２の先頭のデータのアドレスblock_unit_addressは、２×Δｂされ、対応する時刻block_unit_timeは、Ｔ（BU２）とされている。

【００２６】そして、ブロックユニットBU０の時間長delta_block_unit_timeは、ブロックユニットBU１の先頭のデータの時刻Ｔ（BU１）と、ブロックユニットBU０の先頭のデータの時刻Ｔ（BU０）との差（Ｔ（BU１）－Ｔ（BU０））とされている。ブロックユニットBU１の時間長delta_block_unit_timeは、ブロックユニットBU２の先頭のデータの時刻Ｔ（BU２）と、ブロックユニットBU１の先頭のデータの時刻Ｔ（BU１）との差（Ｔ（BU２）－Ｔ（BU１））とされている。さらに、ブロックユニットBU２の時間長（delta_block_unit_time）は、ブロックユニットBU２の最後のデータの時刻end_timeと、ブロックユニットBU２の先頭のデータの時刻Ｔ（BU２）との差（end_time－T(BU2)）とされている。

【００２７】次に、上述のエントリーポイントマップについて説明する。図６に示すトランスポートストリームは、図４に示したトランスポートストリームと同様のト

ランスポートストリームである。ここで斜線で示すトランスポートパケットにおいて、エントリーポイントが開始しているものとする。具体的には、エントリーポイントにおいて、MPEGビデオのシーケンスヘッダとIピクチャデータが開始しているものとする。所定のブロックユニットの中にエントリーポイントが存在する場合、そのブロックユニットのデータの先頭アドレスからエントリーポイントのアドレスまでのオフセットアドレスが計算される。

【００２８】すなわち、図６の例では、ブロックユニットBU０とBU２にエントリーポイント(Iピクチャ）が存在する。そこで、ブロックユニットBU０においては、その先頭のアドレス０から、Iピクチャの先頭のアドレスI_start_addressまでの間隔ａが、オフセットアドレスとして計算される。同時に、ブロックユニットBU２においては、その先頭のアドレス２×Δｂから、Iピクチャの先頭のアドレスI_start_addressまでの間隔ｂがオフセットアドレスとして計算される。

【００２９】図７は、エントリーポイントマップ、すなわちブロックユニット毎のエントリーポイントまでのオフセットアドレスのテーブルの例を示す。entry_point_flagは、対応するブロックユニットBUiにエントリーポイントが存在する時、「１」とされ、存在しない時、「０」とされる。entry_point_flagが「１」であるブロックユニットについて、そのブロックユニットのデータの先頭アドレスblock_unit_addressから、エントリーポイントのアドレスI_start_addressまでのオフセットアドレスI_start_offset_from_block_unit_addressは、次式に示すように計算される。

【００３０】

I_start_offset_from_block_unit_address= I_start_address - block_unit_address

また、エントリーポイント毎に、エントリーポイントのIピクチャデータの終了アドレスI_end_address、エントリーポイントのIピクチャの次のPまたはIピクチャの終了アドレスP１_end_address、エントリーポイントのIピクチャの次の次のPまたはIピクチャの終了アドレスP２_end_addressが、次式に示すように計算される。

【００３１】I_end_offset_address = I_end_address - I_start_address

P１_end_offset_address= P１_end_address - I_start_address

P２_end_offset_address= P２_end_address - I_start_address

これらのアドレスの具体例を図８に示す。図８は、所定のブロックユニットの先頭からはじまるMPEGビデオデータを示す。ここで、I,P,Bはそれぞれ Iピクチャ、Pピクチャ、またはBピクチャを表し、また添え字の数字は、ピクチャの表示順序を示す。このブロックユニットには、I２で示すエントリーポイントのIピクチャが存在する。また、Iピクチャ I２の次のPピクチャはP５であり、Iピクチャ I２の次の次のPピクチャは、P８である。この時、上記式で演算されたI_start_offset_from_block_unit_address, I_end_offset_address,P１_end_offset_address, P２_end_offset_addressは、図に示す関係になる。

【００３２】すなわち、I_end_offset_addressは、Iピクチャ I２の終了アドレスI_end_addressから、Iピクチャ I２の開始アドレスI_start_addressを減算した値とされている。P１_end_offset_addressは、PピクチャP５の終了アドレスP１_end_addressから、Iピクチャ I２の開始アドレスI_start_addressを減算した値とされている。さらにP２_end_offset_addressは、PピクチャP８の終了アドレスP２_end_addressから、Iピクチャ I２の開始アドレスI_start_addressを減算した値とされている。

【００３３】I_start_offset_from_block_unit_addressは、エントリーポイントのアドレスI start_addressから、ブロックユニットのデータの先頭のアドレスblock_unit_addressを減算した値とされている。

【００３４】なお、記録するトランスポートストリームの中に複数のプログラムが含まれる場合、エントリーポイントの情報は、プログラム毎に区別して作成される。また、すべてのプログラムについて、エントリーポイントデータを用意できない場合を考慮して、エントリーポイントマップは、プログラム毎にエントリーポイントデータが存在するかどうかを示す情報(parsed_program_flag) を有する。

【００３５】記録媒体に記録したトランスポートストリームを編集した場合、そのブロックユニットマップは変更（更新）される。次に、その方法を説明する。図９(A)は、図４に示すトランスポートストリームの先頭の２パケットを消去する場合の例を示す。図９(B)は、このようにしてパケットが部分消去された後のトランスポートストリームを示す。図１０は、図９(B)のトランスポートストリームのブロックユニットマップを示す。このようにブロックユニットの途中までのデータが消去された場合、最初のブロックユニットBU０の時間長(first_block_unit_size)が変化するので、これが書き換えられる。図９(B)の場合は、ブロックユニットBU0のデータ長delta_block_unit_timeが、ブロックユニットBU1の先頭アドレスT(BU１)と、消去後のブロックユニットBU0の先頭のパケットｘのタイムスタンプの差分値に変更される。

【００３６】次に上述のブロックユニットマップのシンタックスの例を図１１と図１２に示す。図１１と図１２は、それぞれブロックユニットマップのヘッダ部(BlockUnitMapHeader()) とデータ部(BlockUnitMapData()) を表す。ブロックユニットマップをファイルとして記録する時は、ヘッダ部とデータ部を１つのファイルにして記

録しても良いし、別々のファイルとして記録しても良い。BlockUnitMapHeader()のstart_time, end_timeは、それぞれ、このブロックユニットマップの開始時刻と終了時刻を示し、例えば、あるトランスポートストリームを記録する時の記録開始時刻と記録終了時刻を示す。first_block_unit_sizeは、最初のブロックユニットの時間長を示す。block_unit_size は、第２番目以降のブロックユニットの時間長を示す。number_of_block_unit_entriesは、トランスポートストリームの中のブロックユニットの数を示す。BlockUnitMapData()には、number_of_block_unit_entriesで示される数のdelta_block_unit_address(図５)が書かれる。

【００３７】また、上述のエントリーポイントマップのシンタックスの第１の例を、図１３乃至図１５に示す。図１３は、エントリーポイントマップのヘッダ部(EntryPointMapHeader()) を表し、図１４はエントリーポイントマップのデータ部(EntryPointMapData()) を表す。図１５は、さらに図１４のentry_point_data()のシンタックスを表わしている。エントリーポイントマップをファイルとして記録する時は、ヘッダ部とデータ部を１つのファイルにして記録しても良いし、別々のファイルとして記録しても良い。

【００３８】図１３のEntryPointMapHeader()のnumber_of_programsは、トランスポートストリームの中のプログラム数を示す。このシンタックスの第３行目から第６行目には、記録する各プログラムについて、エントリーマップテーブルが存在するかどうかを示す情報がある。第４行目のprogram_numberは、プログラムを特定（識別）する情報であり、対応するプログラムのPMT(Program Map Table)に書かれている情報である。第５行目のparsed_program_flagは、そのプログラムのエントリーポイントデータが存在するかどうかを示す。

【００３９】第８行目から第１０行目には、記録する各プログラムのPMTの情報が続く。MPEG２_TS_program_map_section()は、記録するトランスポートストリームの中から抽出した、MPEG２ systems規格で規定されているPMTである。ここで、NUMBER_OF_ParsedProgramsは、parsed_program_flagが「１」であるプログラムの数である。第８行目のNUMBER_OF_ParsedProgramsのループの中でデータが現れる順番は、第３行目のnumber_of_programsのループでparsed_program_flag が「１」であるprogram_numberが現れる順番である。

【００４０】図１４のEntryPointMapData()には、記録する各プログラムについてのエントリーポイントのデータが記述される。１つのブロックユニットについてのエントリーポイントのパラメータは、entry_point_flagとentry_point_data()である。１つのブロックユニットについてのentry_point_data()の内容は、図１５に示すように、entry_point_time_stamp, I_start_offset_from_block_unit_address, I_end_offset_address, P１_end_offset_address, P２_end_offset_addressである。ここで、entry_point_time_stampは、エントリーポイントのトランスポートパケットのストリーム上の時刻、または、エントリーポイントのIピクチャのPTS (Presentation Time Stamp) に基づいて計算される。PTSは、MPEG２システムズ規格のPES パケットのヘッダに付加されている情報である。

【００４１】また、上述のエントリーポイントマップのシンタックスの第２の例を、図１６に示す。EntryPointMapHeader()とentry_point_data()の構成は、上述の第１の例における図１３または図１５に示す場合と同様である。この図１６と図１４を比較して明らかなように、各プログラムについてのエントリーポイントのデータの並び方が、図１４の第１の例とは異なる。

【００４２】次に、以下に示す状態の第１の例と第２の例のそれぞれの場合のエントリーマップのデータの並びの例を示す。ここでは、図１７に示すように、トランスポートストリームの中に３個のプログラム(program#１, program#２, program#３)が多重化されていて、ブロックユニットBUi(ｉ＝０, １, ２, ３) 毎に、各プログラムのエントリーポイントがあるものとする。この場合、各パラメータは次のようになる。

【００４３】
number_of_block_unit_entries ＝４
number_of_programs＝３
program_number＝１：parsed_program_flag＝１
program_number＝２：parsed_program_flag＝１
program_number＝３：parsed_program_flag＝１
NUMBER_OF_ParsedPrograms＝３

図１８は、第１の例（図１４の例）の場合のエントリーポイントマップを示す。この場合、プログラム毎にエントリーポイントデータのリストが別れた形になる。すなわち、program#１のEntryPointMapDataは、図１８(A)に示すように、ブロックユニットBU０乃至BU３のそれぞれに、entry_point_dataとして、entry_point_data#１－１乃至entry_point_data#１－４が存在するため、entry_point_flagはそれぞれ「１」とされる。

【００４４】なお、entry_point_data#A-Bは、program_number＝AのB番目のエントリーポイントについてのentry_point_data()を表わす。

【００４５】program#２のEntryPointMapDataは、図１８(B)に示すように、ブロックユニットBU１, BU３には、entry_point_dataが存在しないため、そのentry_point_flagは「０」とされる。これに対して、ブロックユニットBU０, BU２においては、それぞれentry_point_data#２－１, entry_point_data#２－２が存在するため、そのentry_point_flagは「１」とされる。

【００４６】さらに、program#３のEntryPointMapDataのブロックユニットBU０, BU２には、entry_point_dataが存在しないため、entry_point_flagは「０」とされ

る。ブロックユニットBU１，BU３には、entry_point_data#３－１，entry_point_data#３－２がそれぞれ存在するため、そのentry_point_flagは「１」とされている。

【００４７】これらのentry_point_flagと、entry_point_dataが、EntryPointMapDataに記述される。

【００４８】また、図１９は、第２の例（図１６の例）の場合のエントリーポイントマップを示す。

【００４９】この場合、ブロックユニット毎に各プログラムのエントリーポイントデータが時間順に並ぶ形になり、エントリーポイントデータのリストは１つの形となる。すなわち、ブロックユニットBU０において、３つのプログラムprogram#１乃至#３が記述され、それぞれについて、entry_point_flagと対応するentry_point_dataが記述される。この例では、program#３には、entry_point_dataが存在しないため、そのentry_point_flagは「０」とされ、program#１，#２については、entry_point_data#１－１，#２－１が存在するため、そのentry_point_flagは「１」とされている。

【００５０】その他のブロックユニットBU１乃至BU３においても、program#１乃至#３それぞれについて、entry_point_flagと、entry_point_dataが記述される。

【００５１】次に、入力されたトランスポートストリームから、上述のテーブルを作成してトランスポートストリームとともに、記録媒体に記録する動画像記録装置１の構成例を図２０に示す。

【００５２】端子１０から入力されるトランスポートストリームには、１つまたは複数のAVプログラムが多重化されている。端子２２には、ユーザインタフェースによって選択されたAVプログラムのチャネル（サービス名）が入力される。ここで選択されるチャネル数は、１つでも複数でも良い。

【００５３】PIDフィルタ１１は、入力されたトランスポートストリームの中から、ストリーム解析部１２により指定されたPID(Packet ID)のトランスポートパケットを取り出す。PIDは、トランスポートパケットのヘッダの固定位置にある１３ビット長の符号であり、そのトランスポートパケットのペイロードにストアされているデータのタイプを表す。はじめにPIDフィルタ１１は、PID=０x００００であるPAT(Program Association Table)のトランスポートパケットを取り出す。PATには、トランスポートストリームに多重化されている各プログラムのPMT(Program Map Table)のトランスポートパケットのPIDが書かれている。PIDフィルタ１１から出力されるPATのトランスポートパケットは、ストリーム解析部１２へ入力される。

【００５４】カウンタ２４は、記録するトランスポートストリームの先頭パケットから現在のパケットまでのパケット数を計数し、現在のパケットナンバーを、ブロックユニットマップ作成部２３とエントリーポイントマップ作成部１６へ出力する。

【００５５】ストリーム解析部１２は、PCR(Program Clock Reference)を伝送するトランスポートパケットからPCRを抽出して、PLL部１３へ出力する。PCRを伝送するトランスポートパケットのPIDが複数ある場合は、どれか１つのPIDのパケットからPCRが抽出される。PLL部１３は、入力されたPCRに同期して、２７MHzの周波数のクロックを生成し、そのクロックをタイムスタンプ発生部１４に出力する。

【００５６】タイムスタンプ発生部１４は、入力されたクロックをカウントし、そのカウント値に対応したタイムスタンプを生成する。このタイムスタンプは、最初に記録するトランスポートパケットのタイムスタンプをゼロとすれば、そのトランスポートストリームの記録後の経過時間を表すことになる。このタイムスタンプは、ストリーム解析部１２、タイムスタンプ付加部１５、およびブロックユニットマップ作成部２３へ出力される。

【００５７】タイムスタンプ付加部１５は、PIDフィルタ１１から入力されたトランスポートパケットに、その到着時刻を示すタイムスタンプを付加し、タイムスタンプの付加したトランスポートパケットをファイルシステム部１７へ出力する。

【００５８】ブロックユニットマップ作成部２３は、カウンタ２４から入力されるパケットナンバーと、タイムスタンプ発生部１４から入力されるタイムスタンプに基づいて、上述のブロックユニットマップを作成する。作成されたブロックユニットマップは、エントリーポイントマップ作成部１６とファイルシステム部１７へ出力される。

【００５９】ストリーム解析部１２は、プログラム毎の次に示すプログラム情報をエントリーポイントマップ作成部１６へ出力する。

（１）プログラムのprogram_number
（２）プログラムのPMTのトランスポートパケットのPID
（３）プログラムを構成するビデオのトランスポートパケットのPIDとstream_type
（４）プログラムを構成するオーディオのトランスポートパケットのPIDとstream_type
（５）プログラムのPCRのPID

ここで、stream_typeは、PMTに書いてある内容であり、ビデオの場合、MPEG２/MPEG１などのストリームタイプを表し、またオーディオの場合、MPEG１/AC-３などのストリームタイプを表す。

【００６０】ストリーム解析部１２はまた、記録するストリームのエントリーポイントデータを作成し、エントリーポイントマップ作成部１６へ入力する。エントリーポイントデータの内容は、図１５に示すものである。なお、エントリーポイントのタイムスタンプをエントリーポイントのPTSとする場合、PTSはストリーム解析部１２が入力ストリームから取り出すので、タイムスタンプ発生部１４により作成したタイムスタンプをストリーム解

析部１２へ入力する必要はない。

【００６１】エントリーポイントマップ作成部１６は、エントリーポイントデータをプログラム毎にテーブル化し、上述のエントリーポイントマップを作成し、ファイルシステム部１７へ出力する。

【００６２】次に、その動作について説明する。PIDフィルタ１１は、端子１０からトランスポートストリームが入力されると、PID＝０ｘ０００００であるPIDを含むトランスポートパケットを抽出し、ストリーム解析部１２に出力する。ストリーム解析部１２は、この時、図２１のフローチャートに示す処理を実行する。

【００６３】ステップＳ１１で、ストリーム解析部１２は、PIDフィルタ１１からPID＝０ｘ００００のトランスポートパケットを受信すると、そのPATから、端子２２を介して指令された各プログラムのPMTのトランスポートパケットのPIDを取得する。

【００６４】ステップＳ１２で、ストリーム解析部１２は、各プログラムのPMTのPIDをPIDフィルタ１１にセットする。PIDフィルタ１１は、これらPMTのPIDをもつトランスポートパケットを取り出すと、それをストリーム解析部１２へ出力する。

【００６５】ステップＳ１３で、ストリーム解析部１２は、PIDフィルタ１１からPMTのトランスポートパケットを受信する。PMTには、そのプログラムを構成するビデオストリームやオーディオストリームをペイロードに持つトランスポートパケットのPIDやPCR(Program Clock Reference)を伝送しているパケットのPIDが書かれている。ストリーム解析部１２は、ユーザインタフェースによって選択された各プログラムを構成するビデオストリームやオーディオストリームをペイロードに持つトランスポートパケットのPIDとPCRを伝送しているパケットのPIDをここで取得する。

【００６６】ステップＳ１４で、ストリーム解析部１２は、ユーザインタフェースによって選択された各プログラムを構成するビデオストリームやオーディオストリームをペイロードに持つトランスポートパケットのPID とPCRを伝送しているパケットのPIDを、PIDフィルタ１１にセットする。

【００６７】なお、あらかじめEPG(Electrical Program Guide)等を伝送するサービスインフォメーションのパケットのPIDがわかっている場合は、これらのPIDもまた、PIDフィルタ１１にセットされ、それらPIDのパケットも、PIDフィルタ１１から出力される。

【００６８】このようにして、PIDフィルタ１１により抽出されたトランスポートパケットは、カウンタ２４、ストリーム解析部１２およびタイムスタンプ付加部１５に供給される。カウンタ２４は、記録するトランスポートストリームの先頭のパケットから現在のパケットまでのパケット数を計数し、現在のパケットナンバを検知する。検知された現在のパケットNO. は、ブロックユニットマップ作成部２３と、エントリーポイントマップ作成部１６へ供給される。

【００６９】また、ストリーム解析部１２は、入力されるトランスポートパケットからPCRを抽出し、PLL部１３へ供給する。PLL部１３は、入力されたPCRに同期して、２７MHzの周波数のクロックを生成し、タイムスタンプ発生部１４に供給する。

【００７０】タイムスタンプ発生部１４は、入力されたクロックをカウントし、そのカウント値に対応するタイムスタンプを生成する。タイムスタンプ付加部１５は、PIDフィルタ１１から入力されたトランスポートパケットに、その到着時刻を示す、タイムスタンプ発生部１４が発生したタイムスタンプを付加し、ファイルシステム部１７に供給する。

【００７１】ブロックユニットマップ作成部２３は、カウンタ２４から入力されるパケットナンバと、タイムスタンプ発生部１４から入力されるタイムスタンプに基づいて、図５に示したようなブロックユニット毎のblock_unit_addressと、delta_block_unit_timeとを対応させたブロックユニットマップを作成し、エントリーポイントマップ作成部１６と、ファイルシステム部１７へ供給する。

【００７２】ストリーム解析部１２はまた、プログラム毎の上述したプログラム情報を、エントリーポイントマップ作成部１６へ供給する。

【００７３】このため、ストリーム解析部１２は、図２２と図２３に示すような、エントリーポイントの解析処理を実行する。

【００７４】ステップＳ３１でストリーム解析部１２は、記録するプログラムのビデオのPIDと、そのstream_typeをPIDフィルタ１１にセットする。これにより、PIDフィルタ１１から、指定したビデオのパケットが、ストリーム解析部１２に供給される。

【００７５】ステップＳ３２でストリーム解析部１２は、ビデオパケットのポインタvppを初期化し、vpp=０とする。ポインタvppは、現在処理している上記PIDのビデオパケットの順番を表す。

【００７６】ステップＳ３３でストリーム解析部１２は、ビデオパケットのポインタvppをインクリメントする（例えば、１だけ増加する）。

【００７７】ステップＳ３４で、ストリーム解析部１２は、ペイロードの中のストリームに、MPEGビデオのsequence_header_code(３２ビット長で"０ｘ０００００１Ｂ３"の符号)が含まれているか否かを調べる。sequence_header_codeが含まれていない時は、処理はステップＳ３３へ戻る。

【００７８】ステップＳ３４で、ペイロードにsequence_header_codeが含まれていると判定された時は、ステップＳ３５へ進み、ストリーム解析部１２は、sequence_header_codeを含むパケット（最初のIピクチャのパケッ

ト）のアドレスをI_start_addressとする（図８）。

【００７９】ステップＳ３６でストリーム解析部１２は、ビデオパケットのポインタvppをインクリメントする。

【００８０】ステップＳ３７で、ストリーム解析部１２は、上記Iピクチャのデータが終了したか否かを調べる。Iピクチャのデータがまだ終了していない場合、処理はステップＳ３６へ戻る。Iピクチャのデータが終了した場合、処理はステップＳ３８へ進む。

【００８１】ステップＳ３８で、ストリーム解析部１２は、Iピクチャが終了するパケットのアドレスをI_end_addressとする（図８）。以上により、最初のIピクチャのアドレスが決定されたことになる。

【００８２】ストリーム解析部１２は、ステップＳ３９で（ビデオポインタvppはインクリメントしないで）、次のビデオパケットがシーケンスヘッダコードを含んでいるか否かを調べる。パケットがシーケンスヘッダコードを含んでいる場合、処理はステップＳ４７へ進む。パケットがシーケンスヘッダコードを含んでいない場合、処理はステップＳ４０へ進む。

【００８３】ストリーム解析部１２は、ステップＳ４０でビデオパケットのポインタvppをインクリメントする。

【００８４】ストリーム解析部１２は、ステップＳ４１で、Pピクチャまたは Iピクチャが終了したかどうかを調べる。Pピクチャまたは Iピクチャが終了していない場合、処理はステップＳ３９へ戻る。Pピクチャまたは Iピクチャが終了している場合、処理はステップＳ４２へ進む。

【００８５】ストリーム解析部１２は、ステップＳ４２で、Pまたは Iピクチャが終了するパケットのアドレスをP１_end_addressとする（図８）。以上により、Iピクチャの次の最初のPピクチャまたは Iピクチャのアドレスが決定されたことになる。

【００８６】ストリーム解析部１２は、ステップＳ４３で（ビデオポインタvppはインクリメントしないで）、次のビデオパケットがシーケンスヘッダコードを含んでいないか否かを調べる。ビデオパケットがシーケンスヘッダコードを含んでいる場合、処理はステップＳ４７へ進む。ビデオパケットがシーケンスヘッダコードを含んでいない場合、処理はステップＳ４４へ進む。

【００８７】ストリーム解析部１２は、ステップＳ４４でビデオパケットのポインタvppをインクリメントする。

【００８８】ストリーム解析部１２は、ステップＳ４５でPピクチャまたは Iピクチャが終了したかどうかを調べる。Pピクチャまたは Iピクチャが終了していない場合、処理はステップＳ４３へ戻る。Pピクチャまたは Iピクチャが終了している場合、処理はステップＳ４６へ進む。

【００８９】ストリーム解析部１２は、ステップＳ４６で、Pまたは Iピクチャが終了するパケットのアドレスを、P２_end_addressとする（図８）。以上により、Iピクチャの次の次のPピクチャまたは Iピクチャのアドレスが決定されたことになる。

【００９０】ストリーム解析部１２は、ステップＳ４７でI_start_address, I_end_address, P１_end_address, P２_end_addressのアドレスを、エントリーポイントマップ作成部１６へ出力する。なお、この時、P１_end_addressとP２_end_addressの少くとも一方は存在しない場合もある。

【００９１】ストリーム解析部１２は、ステップＳ４８で、現在のパケットが最後の入力パケットであるかどうかを判定する。現在のパケットが最後のパケットでない場合、処理はステップＳ３３へ戻る。現在のパケットが最後のパケットである場合、処理は終了される。

【００９２】以上のビデオストリームの解析は、記録するトランスポートストリームの中に複数のプログラムがある場合は、それぞれのプログラムのビデオパケットに対して行なわれる。

【００９３】ストリーム解析部１２は、以上のようにしてエントリーポイントデータを生成すると、これをエントリーポイントマップ作成部１６に供給する。エントリーポイントマップ作成部１６は、ストリーム解析部１２より供給されたエントリーポイントデータを、プログラム毎にテーブル化し、図７に示すようなエントリーポイントマップを作成し、ファイルシステム部１７に供給する。

【００９４】以上のようにして、ファイルシステム部１７には、タイムスタンプ付加部１５によりタイムスタンプが付加されたトランスポートストリームと、その特徴点を表わす特徴点データとしてのブロックユニットマップと、エントリーポイントマップが、ブロックユニットマップ作成部２３とエントリーポイントマップ作成部１６からそれぞれ供給される。ファイルシステム部１７は、トランスポートストリームと、それに対応する特徴点データをファイル化する。

【００９５】図２４は、このファイル構造の例を表わしている。この例においては、トランスポートストリームファイルの中に、３個のプログラムが多重化されている。同図に示すように、エントリーポイントマップは、ブロックユニットマップに従属する構成とされている。そして、各エントリーポイントマップは、プログラム毎にそれぞれ次のデータを有する。

（１）プログラムのprogram_number

（２）プログラムのPMTのトランスポートパケットのPID

（３）プログラムを構成するビデオのトランスポートパケットのPIDとstream_type

（４）プログラムを構成するオーディオのトランスポートパケットのPIDとstream_type

（５）プログラムのPCRのPID

（６）エントリーポイントのリスト
ファイルシステム部１７により生成されたファイルは、誤り訂正部１８に供給され、誤り訂正符号が付加された後、変調部１９に供給され、所定の方式で変調される。変調部１９より出力された信号は、書き込み部２０に供給され、記録媒体２１に書き込まれる。
【００９６】以上のようにして、トランスポートストリームとその特徴点データが、記録媒体２１に記録される。
【００９７】以上においては、ブロックユニットマップとエントリーポイントマップを、トランスポートストリームから作成するようにしたが、例えば、動画像記録装置自身が、トランスポートストリームを多重化し、生成するような場合、その多重化動作時に、ブロックユニットマップとエントリーポイントマップを、作成するようにすることもできる。図２５は、この場合の構成例を表わしている。
【００９８】すなわち、図２５の例においては、多重化器４０に複数（ｎ個）のプログラムの、ビデオとオーディオのエレメンタリーストリーム#１乃至#nが入力されている。システムタイムクロック部４２は、２７MHzの周波数のシステムタイムクロックをカウントし、タイムスタンプを生成し、コントローラ４１とブロックユニットマップ作成部４３に出力している。コントローラ４１は、多重化器４０に入力された各エレメンタリーストリームを解析し、多重化器４０が、MPEG２システム規格のT-STD(Transport Stream System Target Decoder)を満たして、トランスポートストリームを多重化するように、多重化器４０を制御する。
【００９９】コントローラ４１は、多重化器４０から出力される、トランスポートパケットの数を示すパケットナンバーを、ブロックユニットマップ作成部４３とエントリーポイントマップ作成部４４に出力する。ブロックユニットマップ作成部４３は、コントローラ４１より入力されるパケットナンバと、システムタイムクロック４２より入力されるタイムスタンプに基づいて、ブロックユニットマップを生成する。
【０１００】コントローラ４１はまた、プログラム情報とエントリーポイントデータとを、エントリーポイントマップ作成部４４に出力する。エントリーポイントマップ作成部４４は、コントローラ４１より供給される、パケットナンバ、プログラム情報、およびエントリーポイントデータ、並びにブロックユニットマップ作成部４３より供給されるブロックユニットマップに基づいて、エントリーポイントマップを生成する。
【０１０１】多重化器４０より出力されたトランスポートストリーム、ブロックユニットマップ作成部４３により作成されたブロックユニットマップ、およびエントリーポイントマップ作成部４４により作成されたエントリーポイントマップは、それぞれ、図２０に示したファイルシステム部１７に供給される。ファイルシステム部１７乃至記録媒体２１までの構成は、図２０に示した場合と同様である。
【０１０２】この図２５に示すような構成の動画像記録装置１においては、コントローラ４１が、多重化器４０により多重化されるエレメンタリーストリームから、プログラム情報と、エントリーポイントデータを生成し、エントリーポイントマップ作成部４４に出力する。また、コントローラ４１は、システムタイムクロック４２より入力されるタイムスタンプに対応するパケットナンバを、ブロックユニットマップ作成部４３とエントリーポイントマップ作成部４４に出力する。
【０１０３】ブロックユニットマップ作成部４３は、コントローラ４１から入力されるパケットナンバと、システムタイムクロック４２より入力されるタイムスタンプに基づいて、ブロックユニットマップを作成する。同様に、エントリーポイントマップ作成部４４は、コントローラ４１より入力されるパケットナンバ、プログラム情報、およびエントリーポイントデータ、並びにブロックユニットマップ作成部４３より入力されるブロックユニットマップに基づいて、エントリーポイントマップを作成する。
【０１０４】そして、作成されたトランスポートストリーム、ブロックユニットマップおよびエントリーポイントマップは、図２０に示した場合と同様に、ファイルシステム部１７によりファイル化され、誤り訂正部１８により誤り訂正分が付加される。そして変調部１９によりさらに変調された後、書き込み部２０により、記録媒体２１に記録される。
【０１０５】次に、以上のようにして、トランスポートストリームファイルと、そのストリームの特徴点データが記録された記録媒体２１を再生する動画像再生装置について説明する。図２６は、このような動画像再生装置５１の構成例を表わしている。読み出し部６１は、記録媒体２１に記録されているデータを読み出し、復調部６２に出力する。復調部６２は、読み出し部６１より入力されたデータを復調して、誤り訂正部６３に出力する。誤り訂正部６３は、復調部６２より入力されたデータの誤りを訂正し、ファイルシステム部６４に供給する。
【０１０６】ファイルシステム部６４は、誤り訂正部６３より入力されたデータを、トランスポートストリームファイルと、特徴点データとに分離し、ファイルストリームファイルをデマルチプレクサ６５に供給するとともに、特徴点データを再生制御部６７に出力する。再生制御部６７は、端子６９からユーザインタフェースを介し、てユーザより入力された指令に対応して、読み出し部６１、デマルチプレクサ６５、およびAVデコーダ６６を制御する。
【０１０７】デマルチプレクサ６５は、ファイルシステム部６４より入力されたトランスポートストリームファ

イルから、再生制御部６７からの指令に対応するチャネルのビデオデータと、オーディオデータとを抽出し、AVデコーダ６６に出力する。AVデコーダ６６は、デマルチプレクサ６５より入力された、ビデオデータとオーディオデータをデコードし、端子６８から出力する。

【０１０８】次に、その動作について説明する。記録媒体２１には、図２０（または図２５）の動画像記録装置１で記録したトランスポートストリームファイルと、そのストリームの特徴点データが記録されている。トランスポートストリームファイルには、１つまたは複数のプログラムが多重化されている。

【０１０９】はじめに再生制御部６７は、読み出し部６１に対して、ストリームの特徴点データを読み出すように指示する。このとき、読み出し部６１は、記録媒体２１からストリームの特徴点データを読み出し、復調部６２に出力する。復調部６２は、入力されたデータを復調し、誤り訂正部６３に出力する。誤り訂正部６３は、入力されたデータの誤りを訂正し、ファイルシステム部６４に供給する。ファイルシステム部６４は、入力されたストリーム特徴点データを再生制御部６７に出力する。

【０１１０】端子６９からは、ユーザインタフェースによって再生を指定されたプログラム番号が入力され、それが再生制御部６７へ入力される。再生制御部６７は、そのプログラムのPMTのトランスポートパケットのPID、プログラムを構成するビデオのトランスポートパケットのPIDとstream_type、プログラムを構成するオーディオのトランスポートパケットのPIDとstream_type、並びにPCRのPIDを、特徴点データから読み出し、デマルチプレクサ６５とAVデコーダ６６へ出力する。

【０１１１】さらに、再生制御部６７は、読み出し部６１に対して、トランスポートストリームファイルを読み出すように指示する。この指令に対応して、読み出し部６１は、記録媒体２１からトランスポートストリームファイルを読み出す。このデータは、上述した場合と同様に復調部６２、誤り訂正部６３、ファイルシステム部６４の処理を経て、デマルチプレクサ６５へ入力される。

【０１１２】デマルチプレクサ６５は、ユーザインタフェースにより指定されたプログラムを構成するビデオとオーディオのトランスポートパケットを、入力されたトランスポートストリームから分離し、それをAVデコーダ６６へ入力する。AVデコーダ６６は、ビデオストリームとオーディオストリームを復号し、再生ビデオ信号と再生オーディオ信号として端子６８から出力する。

【０１１３】ユーザインタフェースによってランダムアクセス再生が指示された場合、再生制御部６７は、内部に記憶されているストリームの特徴点データの内容に基づいて、記録媒体２１からのデータの読み出し位置を決定し、ランダムアクセス制御情報を読み出し部６１へ入力する。例えば、ユーザによって選択されたプログラムを所定の時刻から途中再生する場合、再生制御部６７は、ブロックユニットマップに基づいて、指定された時刻に対応するトランスポートストリームのアドレスを計算し、そのアドレスからデータを読み出すように読み出し部６１へ指示する。以下に、その手順を説明する。

【０１１４】ブロックユニットマップのデータからN番目のブロックユニットの先頭データの時刻block_unit_time(N)は、次の様に計算できる。

【０１１５】

【数１】

$$block\_unit\_time(N)=start\_time+\sum_{i=0}^{N-1} delta\_block\_unit\_time(i)$$

ここで、delta_block_unit_time(i)は、i番目のブロックユニットのdelta_block_unit_timeである。i=0のブロックユニットはBUOである。そして、block_unit_time(N)がユーザーから指定された時刻よりも大きくなるNがわかったら、N番目のブロックユニットからデータを読み出せば良いことがわかる。

【０１１６】この場合、記録されているトランスポートストリーム上の0番目のブロックユニットの先頭データのアドレスを０とすれば、N番目(N>0)のブロックユニットの先頭データのアドレスは次のようになる。first_block_unit_size + (N−1)×block_unit_sizeまた、ユーザによって選択されたプログラムに対応するエントリーポイントマップのデータが存在する場合、再生制御部６７は、エントリーポイントデータに基づいて、特殊再生を制御できる。例えば、高速再生の場合、再生制御部６７は、エントリーポイント毎のアドレスのストリームデータを順次連続して読み出すように読み出し部６１へ指示する。

【０１１７】図２７は、この場合の再生制御部６７の動作を表わしている。再生制御部６７は、ステップＳ６１で、ユーザからの指令に対応して、内蔵するメモリに、再生するプログラムのprogram_numberをセットする。

【０１１８】再生制御部６７は、ステップＳ６２で、paesed_program_flagから、そのプログラムのエントリーポイントデータが存在するか否かを調べる。存在する(paesed _program_flag＝１である）場合は、ステップＳ６３へ進む。エントリーポイントデータが存在しない場合は、エントリーポイントマップを使用したデータアクセスはできないので、処理は終了される。

【０１１９】再生制御部６７は、ステップＳ６３で、ユーザにより指定された時刻から読み出し開始するブロックユニットの番号ＢＮを上述のようにして計算する。すなわち、上記した［数１］で計算した値（ブロックユニットの先頭の時刻）が、指定された時刻よりも大きくなるブロックユニットの番号ＢＮが計算される。

【０１２０】再生制御部６７は、ステップＳ６４で、Ｂ

Ｎ番目のブロックユニットに、そのプログラムのエントリーポイントが存在するか否かを、entry_point_flagから調べる。エントリーポイントが存在する(entry_point_flag＝１である）場合は、ステップＳ６５へ進み、存在しない場合は、ステップＳ６７へ進む。

【０１２１】再生制御部６７は、エントリーポイントが存在する場合、ステップＳ６５で、entry_point_data()からエントリーポイントのストリームデータを読み出すアドレスを計算する。ストリームデータの読み出し開始アドレスは、I_start_addressであり、読み出し終了アドレスは、I_end_address、Ｐ１_end_address、またはＰ２_end_addressである。

【０１２２】再生制御部６７は、ステップＳ６６で、ステップＳ６５で計算したアドレスに基づいて、エントリーポイントのストリームデータを読み出すように読み出し部６１に指示する。読み出し部６１はこの指示に対応して読み出し動作を実行する。

【０１２３】再生制御部６７は、ステップＳ６７で、番号ＢＮをインクリメントする。再生制御部６７は、ステップＳ６８で、処理の終了が指令されたか否かを判定し、処理の終了が指令されていない場合は、ステップＳ６４へ戻り、そうでない場合は処理を終了する。

【０１２４】読み出し部６１は、指定されたランダムアクセスポイントからデータを読み出す。読み出されたデータは、復調部６２、誤り訂正部６３、ファイルシステム部６４の処理を経て、デマルチプレクサ６５へ入力され、AVデコーダ６６で復号され、出力される。

【０１２５】このステップＳ６３の計算処理の詳細について、図２８と図２９のフローチャートを参照してさらに説明する。ステップＳ８１において、再生制御部６７に、端子６９からprogram_numberと、再生開始時刻Tstが入力されると、ステップＳ８２において、再生制御部６７は、ステップＳ８１で入力された再生開始時刻Tstが、特徴点データに含まれる、トランスポートストリームの開始時刻start_time（図３(B)）と等しいか否かを判定する。再生開始時刻Tstが開始時刻start_timeと等しい場合には、ステップＳ８６に進み、再生制御部６７は、ブロックユニットの番号を表わす変数Nに０を設定し、そのブロックユニット（０番目のブロックユニット）のblock_unit_address(N)に０を設定する。

【０１２６】これに対して、ステップＳ８２において、再生開始時刻Tstが開始時刻starttimeと等しくないと判定された場合、ステップＳ８３に進み、再生制御部６７は、ブロックユニットマップのヘッダ部を読み込み、ステップＳ８４において、ブロックユニットマップに基づいて、次の不等式を満たす最小の値Ｎを計算する。

【０１２７】Tst≦block_unit_time（Ｎ）

ここで、block_unit_time（Ｎ）は、次式で表される。

【０１２８】

【数２】

$$block\_unit\_time(N)=start\_time+\sum_{i=0}^{N-1} delta\_block\_unit\_time(i)$$

を、次式から演算する。

【０１２９】

$$block\_unit\_address(N) = first\_block\_unit\_size + (N-1)\times block\_unit\_size$$

ステップＳ８５において、再生制御部６７は、Ｎ番目のブロックユニットのアドレスblock_unit_address(N)

Ｎ番目のブロックユニットの先頭データの時刻block_unit_address(N)が求められた時、ステップＳ８７において、再生制御部６７は、Ｎ番目のブロックユニットのアドレスblock_unit_address(N)からのデータ読み出しを、読み出し部６１に指示する。

【０１３０】読み出し部６１は、再生制御部６７からの指令に対応して、ステップＳ８８において、アドレスblock_unit_address(N)からのトランスポートストリームを記録媒体２１から読み出す。読み出されたデータは、復調部６２、誤り訂正部６３、ファイルシステム部６４を介して、デマルチプレクサ６５に供給される。

【０１３１】ステップＳ８９において、再生制御部６７は、デマルチプレクサ６５に対して、ユーザより再生が指示された、プログラムのprogram_numberを出力する。デマルチプレクサ６５は、ステップＳ９０において、再生制御部６７より指示された、program_numberのプログラムのトランスポートパケットを分離し、AVデコーダ６６に出力する。ステップＳ９１において、AVデコーダ６６は、デマルチプレクサ６５より入力されたデータをデコードし、端子６８から出力する。

【０１３２】次に、第２の実施の形態について説明する。この実施の形態の動画像記録装置は、１つまたは複数のプログラムが多重化されているトランスポートストリームをディスク、テープなどの記録媒体に記録する時に、ストリーム上の時間を所定のタイムユニット（単位時間）毎に区切り、タイムユニット毎のデータのストリーム上のアドレスを計算する。そして、このタイムユニット毎のデータのストリーム上のアドレスを示すタイムユニットマップが作成される。さらに、記録するトランスポートストリームのプログラム毎のエントリーポイント（ランダムアクセスポイント）の場所を示すエントリーポイントマップが作成される。エントリーポイントマップは、タイムユニットマップに従属する構造を有する。このタイムユニットマップについて、以下に説明する。

【０１３３】図３０は、複数のAVプログラムが多重化されたトランスポートストリームを示す。ここで横軸は、時間を示し、Δtの間隔のタイムユニットTUi(ｉ＝０，

１，２ ・・・）毎に区切られている。文字TUの後ろに続く数字ｉは、タイムユニットTUの時間順序を示す。最初のオリジナル記録の時は、すべてのタイムユニットTUの時間長は同じ値Δtである。値Δtの大きさは、例えば０．５秒とされる。入力トランスポートストリームの中から１つまたは複数のAVプログラムが記録のために選択される。選択されたトランスポートパケットは斜線を付して示されている。選択されたトランスポートパケットは、一般に図３０(B)に示すように、不規則なタイミングで現れ、Δtの間隔のタイムユニットTUi毎のトランスポートパケットの数は変化する。

【０１３４】選択されたトランスポートパケットは、図３１に示すように、間隔を詰めて記録媒体に記録される。この時、各トランスポートパケットには、それぞれのストリーム上の時刻を示すタイムスタンプが付加される。このタイムスタンプは、例えば、DVフォーマットで規定されているところのトランスポートパケットに付加される４バイト長のTSP_extra_headerと同様のものとされる。

【０１３５】図３１において、横軸は記録されたトランスポートストリームのバイト位置を示すアドレスである。また、横軸上にタイムユニット毎に最初に入力されたトランスポートパケットの先頭アドレスを示す。この例では、タイムユニットTU０，TU１，TU２では、それぞれ４個、３個、または６個のトランスポートパケットが記録されている。２つのタイムユニットにまたがって入力されるトランスポートパケットは、前側のタイムユニットに含められる。タイムユニットTU０，TU１，TU２の最初に入力されたトランスポートパケットの先頭アドレスを、それぞれA(TU０)，A(TU１)，A(TU２)と表わすものとする。

【０１３６】図３２は、タイムユニットマップ、すなわち記録されたトランスポートストリームのタイムユニット毎のデータの先頭アドレスのテーブルの例を示す。ここで、time_unit_addressは、記録されたストリーム上のタイムユニットの先頭データのアドレスを示す。タイムユニットマップでは、タイムユニット毎のデータ長delta_time_unit_addressがテーブル化される。

【０１３７】この例においては、タイムユニットTU０のデータ長は、タイムユニットTU１の先頭のアドレスA(TU１)と、タイムユニットTU０の先頭のアドレスA(TU０)の差(A(TU１)-A(TU０１))で表わされる。同様に、タイムユニットTU１のデータ長は、タイムユニットTU２の先頭のアドレスA(TU２)と、タイムユニットTU１の先頭のアドレスA(TU１)の差(A(TU２)-A(TU１))で表わされ、タイムユニットTU２のデータ長は、タイムユニットTU２の最後のアドレスend_addressと、タイムユニットTU２の先頭のアドレスA(TU２)の差(end_address-A(TU２))で表わされる。

【０１３８】次に、上述のエントリーポイントマップについて説明する。図３３に示すトランスポートストリームは、図３１に示したトランスポートストリームと同様のトランスポートストリームである。ここで斜線で示すトランスポートパケットにおいて、エントリーポイントが開始しているものとする。具体的には、エントリーポイントにおいて、MPEGビデオのシーケンスヘッダとIピクチャデータが開始しているものとする。所定のタイムユニットの中にエントリーポイントが存在する場合、そのタイムユニットのデータの先頭アドレスからエントリーポイントのアドレスまでのオフセットアドレスが計算される。すなわち、図３３の例では、タイムユニットTU０とTU２にエントリーポイント(Iピクチャ）が存在する。そこで、タイムユニットTU０においては、その先頭のアドレスA(TU０)から、Iピクチャの先頭のアドレスI_start_addressまでの間隔ａが、オフセットアドレスとして計算される。同時に、タイムユニットTU２においては、その先頭のアドレスA(TU２)から、Iピクチャの先頭のアドレスI_start_addressまでの間隔ｂがオフセットアドレスとして計算される。

【０１３９】図３４は、エントリーポイントマップ、すなわちタイムユニット毎のエントリーポイントまでのオフセットアドレスのテーブルの例を示す。entry_point_flagは、対応するタイムユニットTUiにエントリーポイントが存在する時、「１」とされ、存在しない時、「０」とされる。entry_point_flagが「１」であるタイムユニットについて、そのタイムユニットのデータの先頭アドレスtime_unit_addressから、エントリーポイントのアドレスI_start_addressまでのオフセットアドレスI_start_offset_from_time_unit_addressは、次式に示すように計算される。

【０１４０】I_start_offset_from_time_unit_address= I_start_address - time_unit_address

また、エントリーポイント毎に、エントリーポイントのIピクチャデータの終了アドレスI_end_address、エントリーポイントのIピクチャの次のPまたはIピクチャの終了アドレスP１_end_address、エントリーポイントのIピクチャの次の次のPまたはIピクチャの終了アドレスP２_end_addressが、次式に示すように計算される。

【０１４１】I_end_offset_address = I_end_address - I_start_address

P１_end_offset_address= P１_end_address - I_start_address

P２_end_offset_address= P２_end_address - I_start_address

これらのアドレスの具体例を図３５に示す。図３５は、所定のタイムユニットの先頭からはじまるMPEGビデオデータを示す。ここで、I,P,Bはそれぞれ Iピクチャ、Pピクチャ、またはBピクチャを表し、また添え字の数字は、ピクチャの表示順序を示す。このタイムユニットには、I２で示すエントリーポイントのIピクチャが存在す

る。また、Ｉピクチャ１２の次のＰピクチャはＰ５であり、Ｉピクチャ１２の次の次のＰピクチャは、Ｐ８である。この時、上記式で演算されたI_start_offset_from_time_unit_address, I_end_offset_address, P１_end_offset_address, P２_end_offset_addressは、図に示す関係になる。

【０１４２】すなわち、I_end_offset_addressは、Ｉピクチャ１２の終了アドレスI_end_addressから、Ｉピクチャ１２の開始アドレスI_start_addressを減算した値とされている。P１_end_offset_addressは、ＰピクチャＰ５の終了アドレスP１_end_addressから、Ｉピクチャ１２の開始アドレスI_start_addressを減算した値とされている。さらにP２_end_offset_addressは、ＰピクチャＰ８の終了アドレスP２_end_addressから、Ｉピクチャ１２の開始アドレスI_start_addressを減算した値とされている。

【０１４３】I_start_offset_from_time_unit_addressは、エントリーポイントのアドレスIstart_addressから、タイムユニットのデータの先頭のアドレスtime_unit_addressを減算した値とされている。

【０１４４】なお、記録するトランスポートストリームの中に複数のプログラムが含まれる場合、エントリーポイントの情報は、プログラム毎に区別して作成される。また、すべてのプログラムについて、エントリーポイントデータを用意できない場合を考慮して、エントリーポイントマップは、プログラム毎にエントリーポイントデータが存在するかどうかを示す情報(parsed_program_flag）を有する。

【０１４５】記録媒体に記録したトランスポートストリームを編集した場合、そのタイムユニットマップは変更（更新）される。次に、その方法を説明する。図３６(A)は、図３１に示すトランスポートストリームの先頭の２パケットと終わりの３パケットを消去する場合の例を示す。図３６(B)は、このようにしてパケットが部分消去された後のトランスポートストリームを示す。図３７は、図３６(B)のトランスポートストリームのタイムユニットマップを示す。このようにタイムユニットの途中までのデータが消去された場合、最初のタイムユニットTU０の時間長(first_time_unit_size)が変化するので、これが書き換えられる。

【０１４６】図３６(B)の場合は、タイムユニットTU０の時間長が、タイムユニットTU１の先頭のパケットPbのタイムスタンプ(A(TU１))と、消去後のタイムユニットTU０の先頭のパケットPaのタイムスタンプ(C)の差分値(A(TU１)－C)に変更される。また、タイムユニットTU２の時間長は、消去後のタイムユニットTU２の最後のパケットのアドレスDと、そのタイムユニットTU２の先頭のパケットのアドレスA(TU２)の差(D－A(TU２))に更新される。タイムユニットマップが変更された場合は、それに関係するエントリーポイントマップも変更される。

【０１４７】次に上述のタイムユニットマップのシンタックスの例を図３８と図３９に示す。図３８と図３９は、それぞれタイムユニットマップのヘッダ部(TimeUnitMapHeader()）とデータ部(TimeUnitMapData()）を表す。タイムユニットマップをファイルとして記録する時は、ヘッダ部とデータ部を１つのファイルにして記録しても良いし、別々のファイルとして記録しても良い。TimeUnitMapHeader()のstart_time, end_time は、それぞれ、このタイムユニットマップの開始時刻と終了時刻を示し、例えば、あるトランスポートストリームを記録する時の記録開始時刻と記録終了時刻を示す。first_time_unit_sizeは、最初のタイムユニットの時間長を示す。time_unit_size は、第２番目以降のタイムユニットの時間長を示す。number_of_time_unit_entriesは、トランスポートストリームの中のタイムユニットの数を示す。TimeUnitMapData()には、number_of_time_unit_entriesで示される数のdelta_time_unit_address(図３２）が書かれる。

【０１４８】また、上述のエントリーポイントマップのシンタックスの第１の例を、図４０と図４１に示す。なお、エントリーポイントマップのヘッダ部(EntryPointMapHeader()）の構成は、図１３に示した場合と同様であるので、ここでは省略する。図４０はエントリーポイントマップのデータ部(EntryPointMapData())を表す。図４１は、さらに図４０のentry_point_data()のシンタックスを表わしている。エントリーポイントマップをファイルとして記録する時は、ヘッダ部とデータ部を１つのファイルにして記録しても良いし、別々のファイルとして記録しても良い。

【０１４９】図４０のEntryPointMapData()には、記録する各プログラムについてのエントリーポイントのデータが記述される。１つのタイムユニットについてのエントリーポイントのパラメータは、entry_point_flagとentry_point_data()である。１つのタイムユニットについてのentry_point_data()の内容は、図４１に示すように、entry_point_time_stamp, I_start_offset_from_time_unit_address, I_end_offset_address, P１_end_offset_address, P２_end_offset_addressである。ここで、entry_point_time_stampは、エントリーポイントのトランスポートパケットのストリーム上の時刻、または、エントリーポイントのＩピクチャのPTS (Presentation Time Stamp) に基づいて計算される。PTSは、MPEG２システムズ規格のPES パケットのヘッダに付加されている情報である。

【０１５０】また、上述のエントリーポイントマップのシンタックスの第２の例は、図１６に示した場合と同様であるので、ここでは省略する。EntryPointMapHeader()とentry_point_data()の構成は、上述の第１の例における図１３または図４１に示す場合と同様である。図１６と図４０を比較して明らかなように、各プログラム

についてのエントリーポイントのデータの並び方が、図４０の第１の例とは異なる。

【０１５１】次に、以下に示す状態の第１の例と第２の例のそれぞれの場合のエントリーマップのデータの並びの例を示す。ここでは、図４２に示すように、トランスポートストリームの中に３個のプログラム(program#１，program#２，program#３)が多重化されていて、タイムユニットTUi(ｉ＝０，１，２，３）毎に、各プログラムのエントリーポイントがあるものとする。この場合、各パラメータは次のようになる。

【０１５２】

number_of_time_unit_entries ＝４

number_of_programs＝３

program_number＝１：parsed_program_flag＝１

program_number＝２：parsed_program_flag＝１

program_number＝３：parsed_program_flag＝１

NUMBER_OF_ParsedPrograms＝３

図４３は、第１の例（図４０の例）の場合のエントリーポイントマップを示す。この場合、プログラム毎にエントリーポイントデータのリストが別れた形になる。すなわち、program#１のEntryPointMapDataは、図４３(A)に示すように、タイムユニットTU０乃至TU３のそれぞれに、entry_point_dataとして、entry_point_data#１－１乃至entry_point_data#１－４が存在するため、entry_point_flagはそれぞれ「１」とされる。

【０１５３】なお、entry_point_data#A-Bは、program_number＝AのB番目のエントリーポイントについてのentry_point_data()を表わす。

【０１５４】program#２のEntryPointMapDataは、図４３(B)に示すように、タイムユニットTU１，TU３には、entry_point_dataが存在しないため、そのentry_point_flagは「０」とされる。これに対して、タイムユニットTU０，TU２においては、それぞれentry_point_data#２－１，entry_point_data#２－２が存在するため、そのentry_point_flagは「１」とされる。

【０１５５】さらに、program#３のEntryPointMapDataのタイムユニットTU０，TU２には、entry_point_dataが存在しないため、entry_point_flagは「０」とされる。タイムユニットTU１，TU３には、entry_point_data#３－１，entry_point_data#３－２がそれぞれ存在するため、そのentry_point_flagは「１」とされている。

【０１５６】これらのentry_point_flagと、entry_point_dataが、EntryPointMapDataに記述される。

【０１５７】また、図４４は、第２の例（図１６の例）の場合のエントリーポイントマップを示す。

【０１５８】この場合、タイムユニット毎に各プログラムのエントリーポイントデータが時間順に並ぶ形になり、エントリーポイントデータのリストは１つの形となる。すなわち、タイムユニットTU０において、３つのプログラムprogram#１乃至#３が記述され、それぞれについて、entry_point_flagと対応するentry_point_dataが記述される。この例では、program#３には、entry_point_dataが存在しないため、そのentry_point_flagは「０」とされ、program#１，#２については、entry_point_data#１－１，#２－１が存在するため、そのentry_point_flagは「１」とされている。

【０１５９】その他のタイムユニットTU１乃至TU３においても、program#１乃至#３それぞれについて、entry_point_flagと、entry_point_dataが記述される。

【０１６０】次に、入力されたトランスポートストリームから、上述のテーブルを作成してトランスポートストリームとともに、記録媒体に記録する動画像記録装置１の構成例を図４５に示す。

【０１６１】この構成例においては、図２０におけるブロックユニットマップ作成部２３が、タイムユニットマップ作成部１２３に変更されている。その他の構成と動作は、図２０乃至図２３における場合と同様であるので、その詳細な説明は省略する。

【０１６２】ストリーム解析部１２は、エントリーポイントデータを生成すると、これをエントリーポイントマップ作成部１６に供給する。エントリーポイントマップ作成部１６は、ストリーム解析部１２より供給されたエントリーポイントデータを、プログラム毎にテーブル化し、図３４に示すようなエントリーポイントマップを作成し、ファイルシステム部１７に供給する。

【０１６３】ファイルシステム部１７には、タイムスタンプ付加部１５によりタイムスタンプが付加されたトランスポートストリームと、その特徴点を表わす特徴点データとしてのタイムユニットマップと、エントリーポイントマップが、タイムユニットマップ作成部１２３とエントリーポイントマップ作成部１６からそれぞれ供給される。ファイルシステム部１７は、トランスポートストリームと、それに対応する特徴点データをファイル化する。

【０１６４】図４６は、このファイル構造の例を表わしている。この例においては、トランスポートストリームファイルの中に、３個のプログラムが多重化されている。同図に示すように、エントリーポイントマップは、タイムユニットマップに従属する構成とされている。各エントリーポイントマップが、プログラム毎に有するデータは、図２４における場合と同様である。

【０１６５】ファイルシステム部１７により生成されたファイルは、誤り訂正部１８に供給され、誤り訂正符号が付加された後、変調部１９に供給され、所定の方式で変調される。変調部１９より出力された信号は、書き込み部２０に供給され、記録媒体２１に書き込まれる。

【０１６６】以上のようにして、トランスポートストリームとその特徴点データが、記録媒体２１に記録される。

【０１６７】以上においては、タイムユニットマップと

エントリーポイントマップを、トランスポートストリームから作成するようにしたが、例えば、動画像記録装置自身が、トランスポートストリームを多重化し、生成するような場合、その多重化動作時に、タイムユニットマップとエントリーポイントマップを、作成するようにすることもできる。図４７は、この場合の構成例を表わしている。

【０１６８】図４７の例においては、図２５の例におけるブロックユニットマップ作成部４３が、タイムユニットマップ作成部１４３に変更されている。その他の構成と動作は、図２５における場合と同様であるので、ここでは省略する。

【０１６９】以上のようにして、トランスポートストリームファイルと、そのストリームの特徴点データが記録された記録媒体２１を再生する動画像再生装置は、図２６に示した場合と同様となる。

【０１７０】次に、その動作について説明する。記録媒体２１には、図４５（または図４７）の動画像記録装置１で記録したトランスポートストリームファイルと、そのストリームの特徴点データが記録されている。トランスポートストリームファイルには、１つまたは複数のプログラムが多重化されている。

【０１７１】はじめに再生制御部６７は、読み出し部６１に対して、ストリームの特徴点データを読み出すように指示する。このとき、読み出し部６１は、記録媒体２１からストリームの特徴点データを読み出し、復調部６２に出力する。復調部６２は、入力されたデータを復調し、誤り訂正部６３に出力する。誤り訂正部６３は、入力されたデータの誤りを訂正し、ファイルシステム部６４に供給する。ファイルシステム部６４は、入力されたストリーム特徴点データを再生制御部６７に出力する。

【０１７２】端子６９からは、ユーザインタフェースによって再生を指定されたプログラム番号が入力され、それが再生制御部６７へ入力される。再生制御部６７は、そのプログラムのPMTのトランスポートパケットのPID、プログラムを構成するビデオのトランスポートパケットのPIDとstream_type、プログラムを構成するオーディオのトランスポートパケットのPIDとstream_type、並びにPCRのPIDを、特徴点データから読み出し、デマルチプレクサ６５とAVデコーダ６６へ出力する。

【０１７３】さらに、再生制御部６７は、読み出し部６１に対して、トランスポートストリームファイルを読み出すように指示する。この指令に対応して、読み出し部６１は、記録媒体２１からトランスポートストリームファイルを読み出す。このデータは、上述した場合と同様に復調部６２、誤り訂正部６３、ファイルシステム部６４の処理を経て、デマルチプレクサ６５へ入力される。

【０１７４】デマルチプレクサ６５は、ユーザインタフェースにより指定されたプログラムを構成するビデオとオーディオのトランスポートパケットを、入力されたトランスポートストリームから分離し、それをAVデコーダ６６へ入力する。AVデコーダ６６は、ビデオストリームとオーディオストリームを復号し、再生ビデオ信号と再生オーディオ信号として端子６８から出力する。

【０１７５】ユーザインタフェースによってランダムアクセス再生が指示された場合、再生制御部６７は、内部に記憶されているストリームの特徴点データの内容に基づいて、記録媒体２１からのデータの読み出し位置を決定し、ランダムアクセス制御情報を読み出し部６１へ入力する。例えば、ユーザによって選択されたプログラムを所定の時刻から途中再生する場合、再生制御部６７は、タイムユニットマップに基づいて、指定された時刻に対応するトランスポートストリームのアドレスを計算し、そのアドレスからデータを読み出すように読み出し部６１へ指示する。以下に、その手順を説明する。

【０１７６】ゼロ番目のタイムユニットTU０の先頭データの時刻をstart_timeとすれば、N番目(N>０)のタイムユニットの先頭データの時刻は、(start_time+first_time_unit_size+(N－１)＊ time_unit_size)となる。ユーザから指定された時刻よりもタイムユニットの先頭データの時刻が大きくなるタイムユニットの番号がわかったら、その番号のタイムユニットからデータを読み出せば良いことがわかる。

【０１７７】この場合、記録されたストリーム上の０番目のタイムユニットの先頭データのアドレスを０とすれば、N番目のタイムユニットの先頭データのアドレスtime_unit_address(N)は、次の様に計算できる。

【０１７８】

【数３】

$$time\_unit\_address(N)=\sum_{i=0}^{N-1} delta\_time\_unit\_address(i)$$

また、ユーザによって選択されたプログラムに対応するエントリーポイントマップのデータが存在する場合、再生制御部６７は、エントリーポイントデータに基づいて、特殊再生を制御できる。例えば、高速再生の場合、再生制御部６７は、エントリーポイント毎のアドレスのストリームデータを順次連続して読み出すように読み出し部６１へ指示する。

【０１７９】図４８は、この場合の再生制御部６７の動作を表わしている。再生制御部６７は、ステップＳ１６１で、ユーザからの指令に対応して、内蔵するメモリに、再生するプログラムのprogram_numberをセットする。

【０１８０】再生制御部６７は、ステップＳ１６２で、paesed_program_flagから、そのプログラムのエントリーポイントデータが存在するか否かを調べる。存在する(paesed _program_flag＝１である）場合は、ステップ

Ｓ１６３へ進む。エントリーポイントデータが存在しない場合は、エントリーポイントマップを使用したデータアクセスはできないので、処理は終了される。

【０１８１】再生制御部６７は、ステップＳ１６３で、ユーザにより指定された時刻から読み出し開始するタイムユニットの番号TNを上述のようにして計算する。すなわち、start_time＋first_time_unit_size＋(N－１)*time_unit_sizeの値（タイムユニットの先頭の時刻）が、指定された時刻よりも大きくなるタイムユニットの番号TNが計算される。

【０１８２】再生制御部６７は、ステップＳ１６４で、TN番目のタイムユニットに、そのプログラムのエントリーポイントが存在するか否かを、entry_point_flagから調べる。エントリーポイントが存在する(entry_point_flag＝１である）場合は、ステップＳ１６５へ進み、存在しない場合は、ステップＳ１６７へ進む。

【０１８３】再生制御部６７は、エントリーポイントが存在する場合、ステップＳ１６５で、entry_point_data()からエントリーポイントのストリームデータを読み出すアドレスを計算する。ストリームデータの読み出し開始アドレスは、I_start_addressであり、読み出し終了アドレスは、I_end_address、P１_end_address、またはP２_end_addressである。

【０１８４】再生制御部６７は、ステップＳ１６６で、ステップＳ１６５で計算したアドレスに基づいて、エントリーポイントのストリームデータを読み出すように読み出し部６１に指示する。読み出し部６１はこの指示に対応して読み出し動作を実行する。

【０１８５】再生制御部６７は、ステップＳ１６７で、番号TNをインクリメントする。再生制御部６７は、ステップＳ１６８で、処理の終了が指令されたか否かを判定し、処理の終了が指令されていない場合は、ステップＳ１６４へ戻り、そうでない場合は処理を終了する。

【０１８６】読み出し部６１は、指定されたランダムアクセスポイントからデータを読み出す。読み出されたデータは、復調部６２、誤り訂正部６３、ファイルシステム部６４の処理を経て、デマルチプレクサ６５へ入力され、AVデコーダ６６で復号され、出力される。

【０１８７】このステップＳ１６３の計算処理の詳細について、図４９と図５０のフローチャートを参照してさらに説明する。ステップＳ１８１において、再生制御部６７に、端子６９からprogram_numberと、再生開始時刻Tstが入力されると、ステップＳ１８２において、再生制御部６７は、ステップＳ１８１で入力された再生開始時刻Tstが、特徴点データに含まれる、トランスポートストリームの開始時刻start_time（図３０(B)）と等しいか否かを判定する。再生開始時刻Tstが開始時刻start_timeと等しい場合には、ステップＳ１８６に進み、再生制御部６７は、タイムユニットの番号を表わす変数Nに０を設定し、そのタイムユニット（０番目のタイムユニット）のtime_unit_address(N)に０を設定する。

【０１８８】これに対して、ステップＳ１８２において、再生開始時刻Tstが開始時刻starttimeと等しくないと判定された場合、ステップＳ１８３に進み、再生制御部６７は、タイムユニットマップのヘッダ部を読み込み、ステップＳ１８４において、次の不等式を満たす最小の値Ｎを計算する。

【０１８９】Tst≦start_time＋first_time_unit_size＋(N－１)×time_unit_size

ステップＳ１８５において、再生制御部６７は、タイムユニットマップのデータに基づいて、数１で示す式に従って、time_unit_address(N)を演算する。

【０１９０】Ｎ番目のタイムユニットの先頭データの時刻time_unit_address(N)が求められた時、ステップＳ１８７において、再生制御部６７は、Ｎ番目のタイムユニットのアドレスtime_unit_address(N)からのデータ読み出しを、読み出し部６１に指示する。

【０１９１】読み出し部６１は、再生制御部６７からの指令に対応して、ステップＳ１８８において、アドレスtime_unit_address(N)からのトランスポートストリームを記録媒体２１から読み出す。読み出されたデータは、復調部６２、誤り訂正部６３、ファイルシステム部６４を介して、デマルチプレクサ６５に供給される。

【０１９２】ステップＳ１８９において、再生制御部６７は、デマルチプレクサ６５に対して、ユーザより再生が指示された、プログラムのprogram_numberを出力する。デマルチプレクサ６５は、ステップＳ１９０において、再生制御部６７より指示された、program_numberのプログラムのトランスポートパケットを分離し、AVデコーダ６６に出力する。ステップＳ１９１において、AVデコーダ６６は、デマルチプレクサ６５より入力されたデータをデコードし、端子６８から出力する。

【０１９３】上述した一連の処理は、ハードウエアにより実行させることもできるが、ソフトウエアにより実行させることもできる。一連の処理をソフトウエアにより実行させる場合には、そのソフトウエアを構成するプログラムが、専用のハードウエアとしての動画像記録再生装置に組み込まれているコンピュータ、または、各種のプログラムをインストールすることで、各種の機能を実行することが可能な、例えば汎用のパーソナルコンピュータなどにインストールされる。

【０１９４】次に、図５１を参照して、上述した一連の処理を実行するプログラムをコンピュータにインストールし、コンピュータによって実行可能な状態とするために用いられる媒体について、そのコンピュータが汎用のパーソナルコンピュータである場合を例として説明する。

【０１９５】プログラムは、図５１（Ａ）に示すように、コンピュータ３０１に内蔵されている記録媒体としてのハードディスク３０２や半導体メモリ３０３に予め

インストールした状態でユーザに提供することができる。

【０１９６】あるいはまた、プログラムは、図５１（Ｂ）に示すように、フロッピーディスク３１１、CD-ROM(Compact Disk-Read Only Disk) ３１２、MO(Magneto-Optical)ディスク３１３、DVD(Digital Versatile Disk)３１４、磁気ディスク３１５、半導体メモリ３１６などの記録媒体に、一時的あるいは永続的に格納し、パッケージソフトウエアとして提供することができる。

【０１９７】さらに、プログラムは、図５１（Ｃ）に示すように、ダウンロードサイト３２１から、デジタル衛星放送用の人工衛星３２２を介して、コンピュータ３２３に無線で転送したり、ローカルエリアネットワーク、インターネットといったネットワーク１３１を介して、コンピュータ３２３に有線で転送し、コンピュータ３２３において、内蔵するハードディスクなどに格納させることができる。

【０１９８】本明細書における媒体とは、これら全ての媒体を含む広義の概念を意味するものである。

【０１９９】また、本明細書において、媒体により提供されるプログラムを記述するステップは、記載された順序に沿って時系列的に行われる処理はもちろん、必ずしも時系列的に処理されなくとも、並列的あるいは個別に実行される処理をも含むものである。

【０２００】なお、本明細書において、システムとは、複数の装置により構成される装置全体を表すものである。

【０２０１】このように、１つまたは複数のトランスポートストリームが記録された記録媒体にランダムアクセスして再生する場合、Iピクチャやオーディオフレームの開始位置を効率よくサーチすることができるので、ユーザ入力に対して応答の速いランダムアクセス再生をすることができる。

【０２０２】

【発明の効果】以上の如く、請求項１に記載のデータ処理装置、請求項８に記載のデータ処理方法、および請求項９に記載の媒体によれば、符号化ストリームをユニット化し、ユニット毎のデータの量を表すデータ量情報を含むユニットマップを作成するようにしたので、応答の速いランダムアクセスが可能となる。

【図面の簡単な説明】

【図１】従来のトランスポートストリームのパケットを説明する図である。

【図２】従来の記録されるトランスポートストリームを説明する図である。

【図３】本発明のトランスポートストリームを説明する図である。

【図４】本発明の記録されるトランスポートストリームを説明する図である。

【図５】ブロックユニットマップの例を示す図である。

【図６】ブロックユニット毎のオフセットアドレスを説明する図である。

【図７】エントリーポイントマップの例を示す図である。

【図８】エントリーポイントデータを説明する図である。

【図９】データの消去を説明する図である。

【図１０】データを消去した時のブロックユニットマップの例を示す図である。

【図１１】BlockUnitMapHeader()のシンタックスを示す図である。

【図１２】BlockUnitMapData()のシンタックスを示す図である。

【図１３】EntryPointMapHeader()のシンタックスを示す図である。

【図１４】EntryPointMapData()のシンタックスを示す図である。

【図１５】entry point data()のシンタックスを示す図である。

【図１６】EntryPointMapData()のシンタックスを示す図である。

【図１７】トランスポートストリームファイルのエントリーポイントを説明する図である。

【図１８】EntryPointMapDataの例を示す図である。

【図１９】EntryPointMapDataの例を示す図である。

【図２０】本発明を適用した動画像記録装置の構成例を示すブロック図である。

【図２１】図２０の動画像記録装置の動作を説明するフローチャートである。

【図２２】図２０の動画像記録装置の動作を説明するフローチャートである。

【図２３】図２０の動画像記録装置の動作を説明するフローチャートである。

【図２４】トランスポートストリームファイルのブロックユニットマップとエントリーポイントマップの関係を説明する図である。

【図２５】本発明を適用した動画像記録装置の他の構成例を示すブロック図である。

【図２６】本発明を適用した動画像再生装置の構成例を示すブロック図である。

【図２７】図２６の動画像再生装置の動作を説明するフローチャートである。

【図２８】図２６の動画像再生装置の動作を説明するフローチャートである。

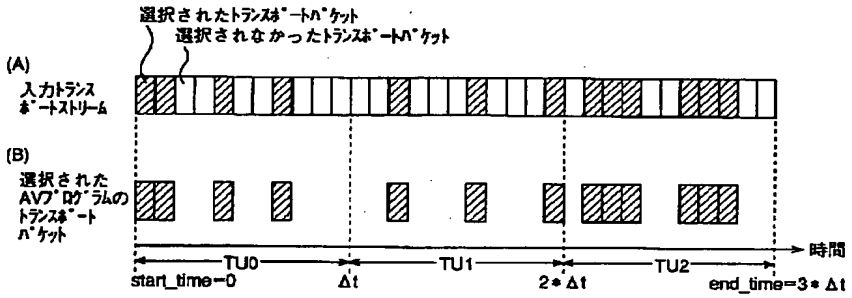
【図２９】図２６の動画像再生装置の動作を説明するフローチャートである。

【図３０】本発明のトランスポートストリームを説明する図である。

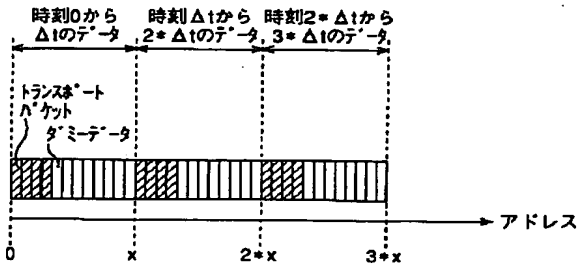
【図３１】本発明の記録されるトランスポートストリームを説明する図である。

【図３２】タイムユニットマップの例を示す図である。
【図３３】タイムユニット毎のオフセットアドレスを説明する図である。
【図３４】エントリーポイントマップの例を示す図である。
【図３５】エントリーポイントデータを説明する図である。
【図３６】データの消去を説明する図である。
【図３７】データを消去した時のタイムユニットマップの例を示す図である。
【図３８】TimeUnitMapHeader()のシンタックスを示す図である。
【図３９】TimeUnitMapData()のシンタックスを示す図である。
【図４０】EntryPointMapData()のシンタックスを示す図である。
【図４１】entry point data()のシンタックスを示す図である。
【図４２】トランスポートストリームファイルのエントリーポイントを説明する図である。
【図４３】EntryPointMapDataの例を示す図である。
【図４４】EntryPointMapDataの例を示す図である。
【図４５】本発明を適用した動画像記録装置の構成例を示すブロック図である。
【図４６】トランスポートストリームファイルのタイムユニットマップとエントリーポイントマップの関係を説明する図である。
【図４７】本発明を適用した動画像記録装置の他の構成例を示すブロック図である。
【図４８】図２６の動画像再生装置の動作を説明するフローチャートである。
【図４９】図２６の動画像再生装置の動作を説明するフローチャートである。
【図５０】図２６の動画像再生装置の動作を説明するフローチャートである。
【図５１】媒体を説明する図である。
【符号の説明】
１ 動画像記録装置，　１１ PIDフィルタ，　１２ ストリーム解析部，１４ タイムスタンプ発生部，　１５ タイムスタンプ付加部，　１６ エントリーポイントマップ作成部，　１７ ファイルシステム部，　２１ 記録媒体，２３ タイムユニットマップ作成部，　２４ カウンタ，　４０ 多重化器，４１ コントローラ，　４２ システムタイムクロック部，　４３ ブロックユニットマップ作成部，　４４エントリーポイントマップ作成部，　６１ 読み出し部，　６５ デマルチプレクサ，　６６ AVデコーダ，　６７ 再生制御部

【図１】



【図２】



【図３７】

タイムユニット	delta_time_unit_address
TU0	A(TU1)−C
TU1	A(TU2)−A(TU1)
TU2	D−A(TU2)

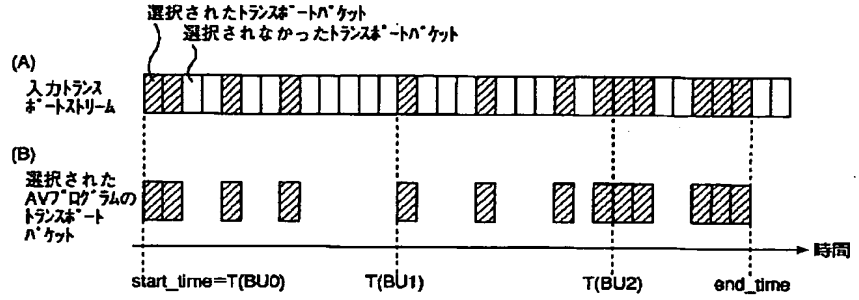
部分消去後のタイムユニットマップ

【図５】

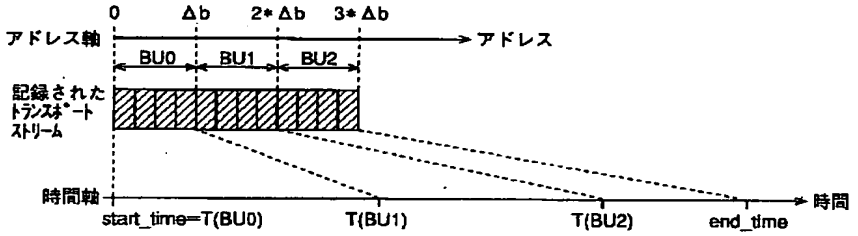
ブロックユニット	block_unit_address	block_unit_time	delta_block_unit_time
BU0	0	T(BU0)	T(BU1)−T(BU0)
BU1	Δb	T(BU1)	T(BU2)−T(BU1)
BU2	2*Δb	T(BU2)	end_time−T(BU2)

ブロックユニットマップ

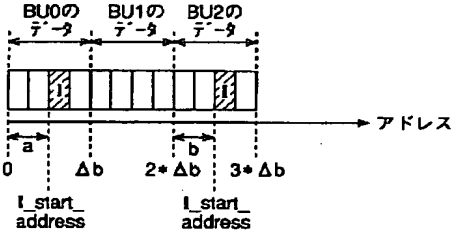
【図３】



【図４】



【図６】

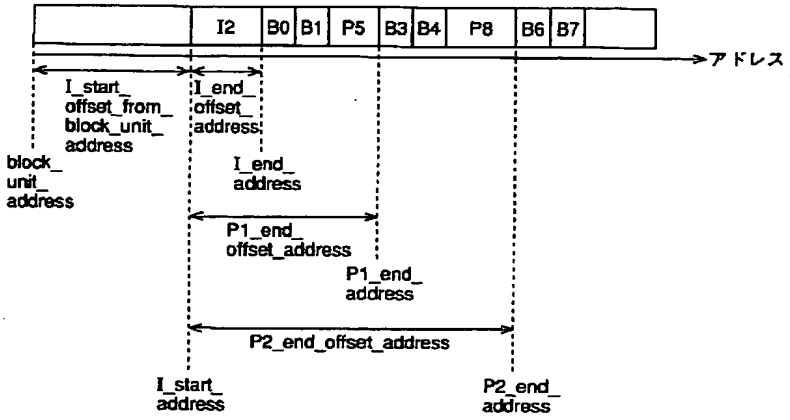


【図７】

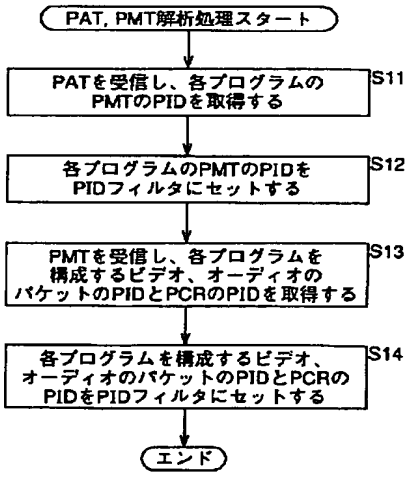
ブロックユニット	entry_point_flag	I_start_offset_from_block_unit_address
BU0	1	a
BU1	0	–
BU2	1	b

エントリポイントマップ

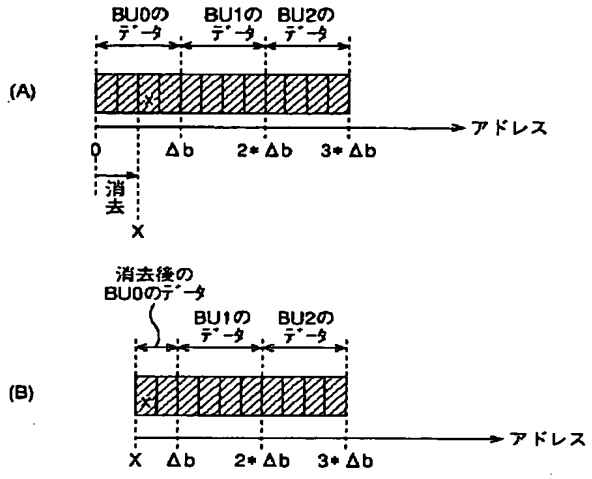
【図８】



【図２１】



【図９】



【図１０】

ﾌﾞﾛｯｸﾕﾆｯﾄ	block_unit_time	delta_block_unit_time
BU0	ﾊﾟｹｯﾄxのﾀｲﾑｽﾀﾝﾌﾟ	T(BU1)−ﾊﾟｹｯﾄxのﾀｲﾑｽﾀﾝﾌﾟ
BU1	T(BU1)	T(BU2)−T(BU1)
BU2	T(BU2)	end_time−T(BU2)

部分消去後のブロックユニットマップ

【図１１】

BlockUnitMapHeader()

	Syntax	Bits
1	BlockUnitMapHeader() {	
2	start_time	N1
3	end_time	N2
4	first_block_unit_size	N3
5	block_unit_size	N4
6	number_of_block_unit_entries	N5
7	}	

【図１２】

BlockUnitMapData()

	Syntax	Bits
1	BlockUnitMapData() {	
2	for(i=0; i<number_of_block_unit_entries; i++) {	
3	delta_block_unit_address	N6
4	}	
5	}	

【図１３】

EntryPointMapHeader()

	Syntax	Bits
1	EntryPointMapsHeader() {	
2	number_of_programs	K1
3	for(i=0; i<number_of_programs; i++) {	
4	program_number	16
5	parsed_program_flag	1
6	}	
7		
8	for(i=0; i<NUMBER_OF_ParsedPrograms; i++) {	
9	MPEG_TS_program_map_section()	
10	}	
11	}	

【図１４】

EntryPointMapData()

	Syntax	Bits
1	EntryPointMapData() {	
2	for(i=0; i<NUMBER_OF_ParsedPrograms; i++) {	
3	for(j=0; j<number_of_block_unit_entries; j++)	
4	entry_point_flag	1
5	entry_point_data()	
6	}	
7	}	

【図１５】

entry_point_data()

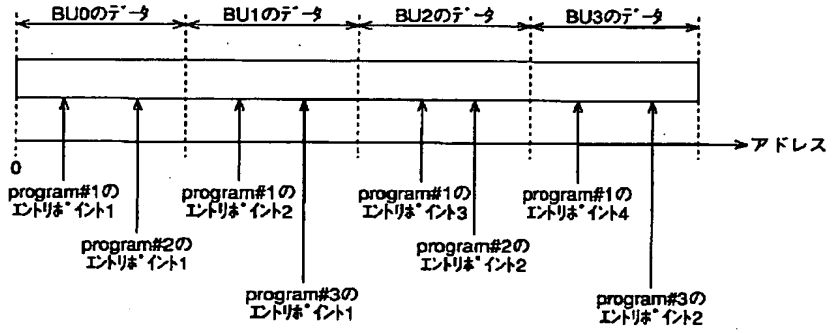
	Syntax	Bits
1	entry_point_data() {	
2	entry_point_time_stamp	K2
3	I_start_offset_from_block_unit_address	K3
4	I_end_offset_address	K4
5	P1_end_offset_address	K5
6	P2_end_offset_address	K6
7	}	

【図１６】

EntryPointMapData()

	Syntax	Bits
1	EntryPointMapData() {	
2	for(i=0; i<number_of_block_unit_entries; i++) {	
3	for(j=0; j<NUMBER_OF_ParsedPrograms; j++)	
4	entry_point_flag	1
5	entry_point_data()	
6	}	
7	}	
8	}	

【図１７】



【図１８】

program#1のEntryPointMapData

(A)

ブロックユニット	entry_point_flag	entry_point_data
BU0	1	entry_point_data#1−1
BU1	1	entry_point_data#1−2
BU2	1	entry_point_data#1−3
BU3	1	entry_point_data#1−4

program#2のEntryPointMapData

(B)

ブロックユニット	entry_point_flag	entry_point_data
BU0	1	entry_point_data#2−1
BU1	0	−(dummy data)
BU2	1	entry_point_data#2−2
BU3	0	−

program#3のEntryPointMapData

(C)

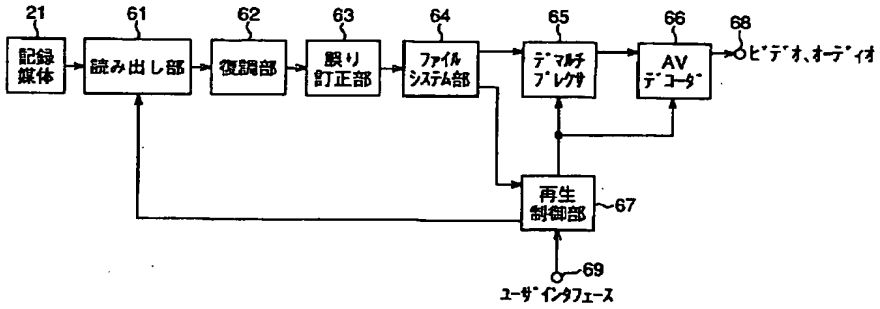
ブロックユニット	entry_point_flag	entry_point_data
BU0	0	−
BU1	1	entry_point_data#3−1
BU2	0	−
BU3	1	entry_point_data#3−2

【図１９】

program#1, program#2, program#3のEntryPointMapData

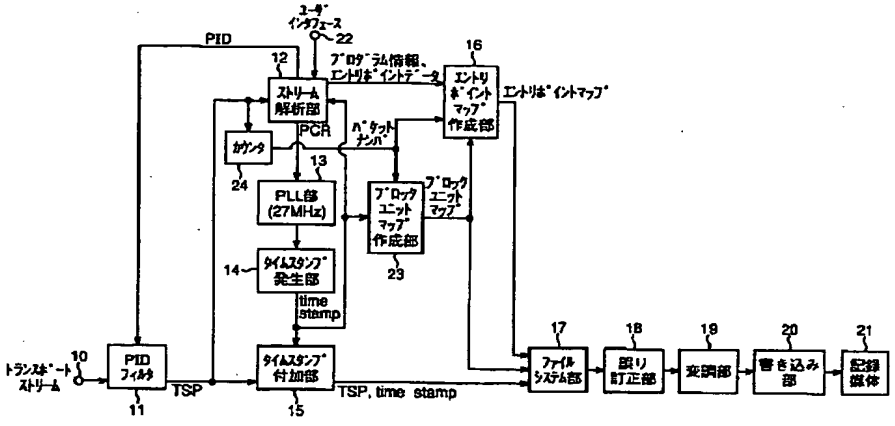
ブロックユニット	Program number	entry_point_flag	entry_point_data
BU0	#1	1	entry_point_data#1−1
	#2	1	entry_point_data#2−1
	#3	0	−
BU1	#1	1	entry_point_data#1−2
	#2	0	−
	#3	1	entry_point_data#3−1
BU2	#1	1	entry_point_data#1−3
	#2	1	entry_point_data#2−2
	#3	0	−
BU3	#1	1	entry_point_data#1−4
	#2	0	−
	#3	1	entry_point_data#3−2

【図２６】



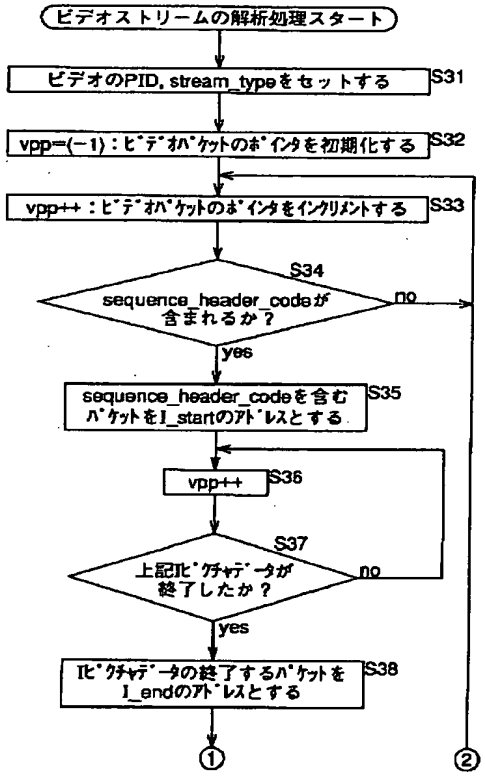
動画像再生装置 51

【図２０】

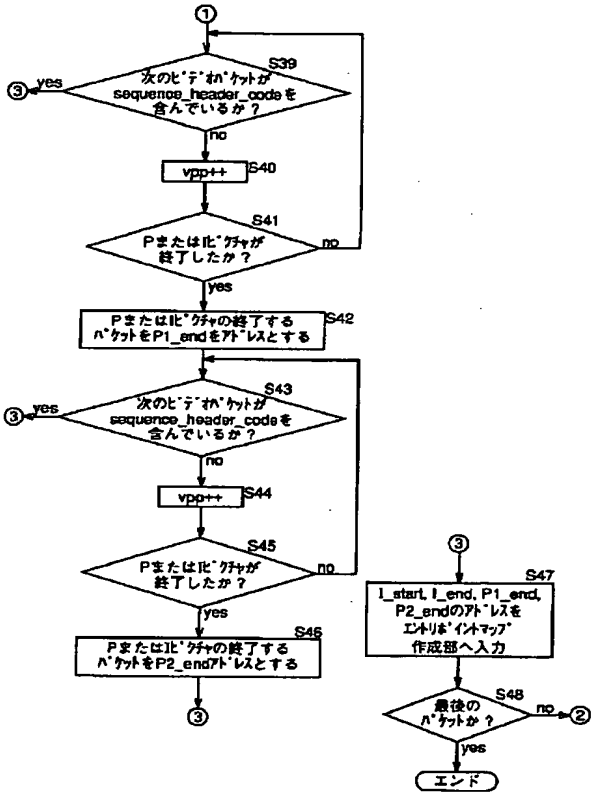


動画像記録装置１

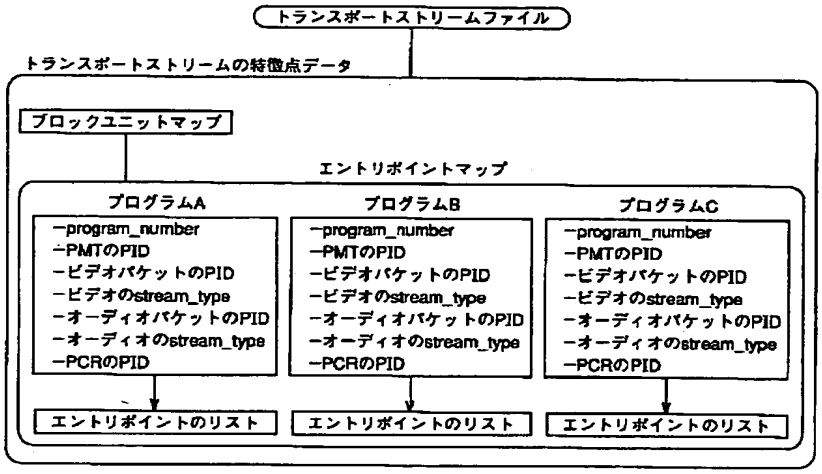
【図２２】



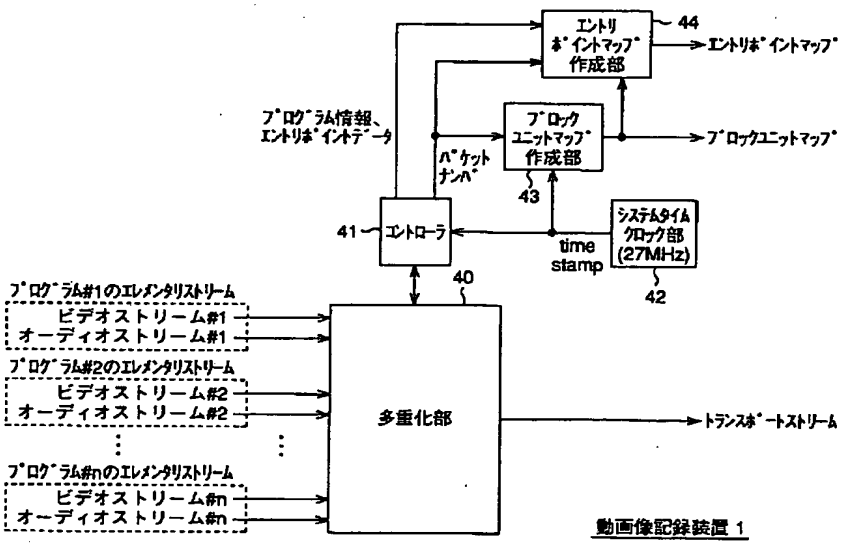
【図２３】



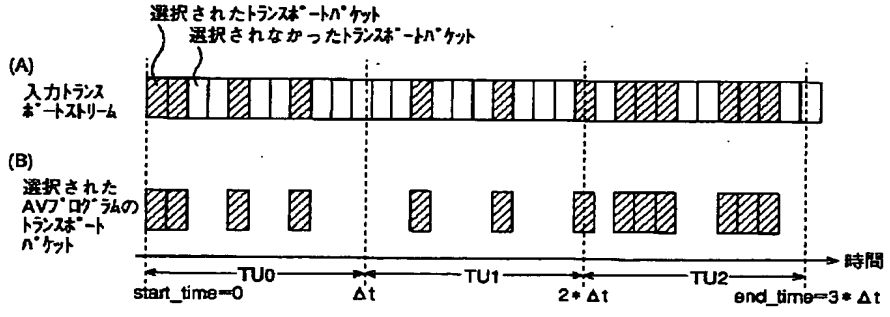
【図２４】


トランスポートストリームファイル
トランスポートストリームの特徴点データ
ブロックユニットマップ
エントリポイントマップ
プログラムA
−program_number
−PMTのPID
−ビデオパケットのPID
−ビデオのstream_type
−オーディオパケットのPID
−オーディオのstream_type
−PCRのPID
プログラムB
−program_number
−PMTのPID
−ビデオパケットのPID
−ビデオのstream_type
−オーディオパケットのPID
−オーディオのstream_type
−PCRのPID
プログラムC
−program_number
−PMTのPID
−ビデオパケットのPID
−ビデオのstream_type
−オーディオパケットのPID
−オーディオのstream_type
−PCRのPID
エントリポイントのリスト
エントリポイントのリスト
エントリポイントのリスト

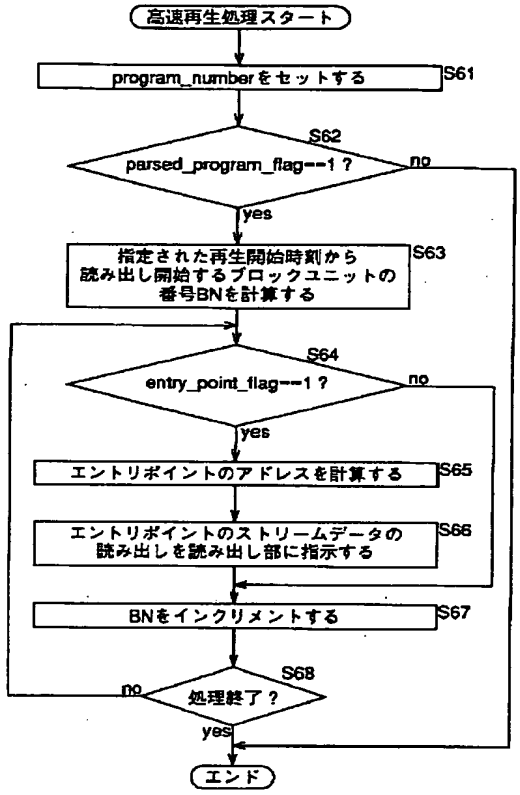
【図２５】


エントリポイントマップ作成部
44
エントリポイントマップ
プログラム情報、エントリポイントデータ
パケットナンバ
ブロックユニットマップ作成部
ブロックユニットマップ
43
41
コントローラ
time stamp
システムタイムクロック部(27MHz)
42
40
プログラム#1のエレメンタリストリーム
ビデオストリーム#1
オーディオストリーム#1
プログラム#2のエレメンタリストリーム
ビデオストリーム#2
オーディオストリーム#2
プログラム#nのエレメンタリストリーム
ビデオストリーム#n
オーディオストリーム#n
多重化部
トランスポートストリーム
動画像記録装置 1

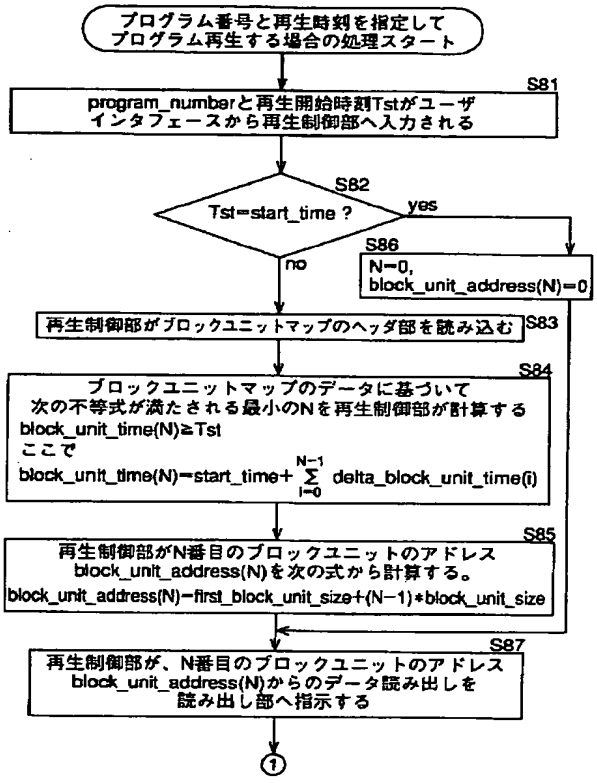
【図３０】


選択されたトランスポートパケット
選択されなかったトランスポートパケット
(A)
入力トランスポートストリーム
(B)
選択されたAVプログラムのトランスポートパケット
時間
TU0
TU1
TU2
start_time=0
Δt
2*Δt
end_time=3*Δt

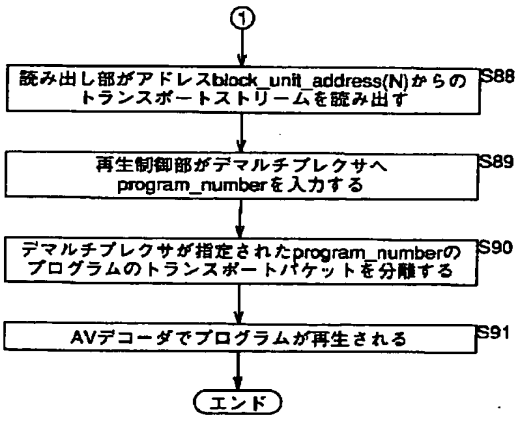
【図２７】



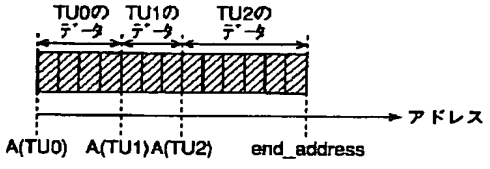
【図２８】



【図２９】



【図３１】

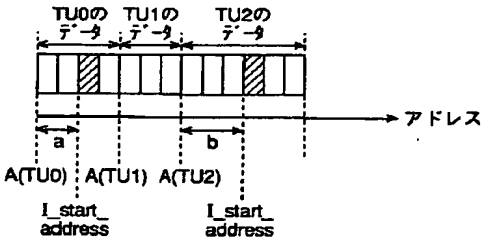


【図３２】

タイムユニット	time_unit_address	delta_time_unit_address
TU0	A(TU0)	A(TU1)−A(TU0)
TU1	A(TU1)	A(TU2)−A(TU1)
TU2	A(TU2)	end_address−A(TU2)

タイムユニットマップ

【図３３】

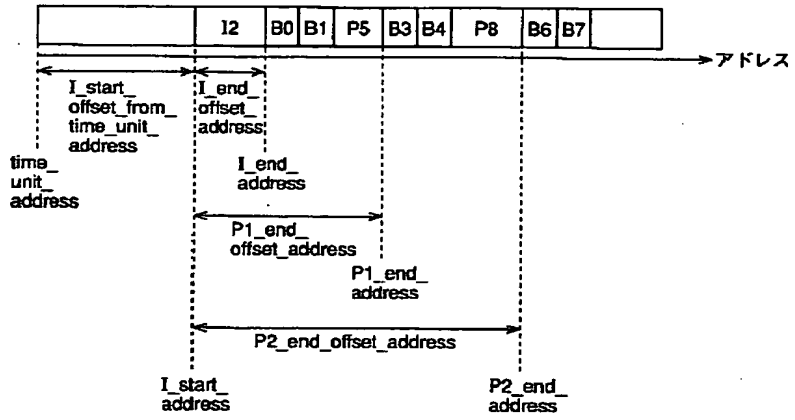


【図３４】

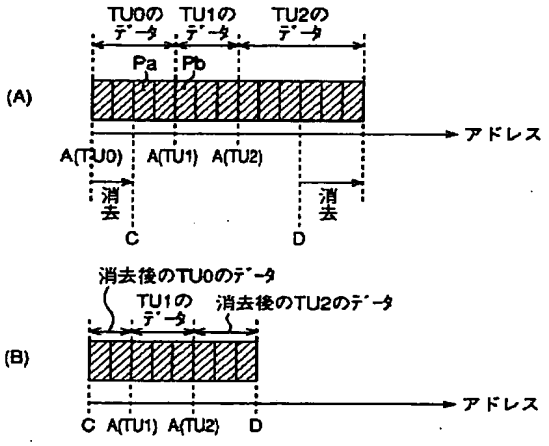
タイムユニット	entry_point_flag	I_start_offset_from_time_unit_address
TU0	1	a
TU1	0	–
TU2	1	b

エントリポイントマップ

【図３５】



【図３６】



【図３８】

TimeUnitMapHeader()

	Syntax	Bits
1	TimeUnitMapHeader() {	
2	start_time	N1
3	end_time	N2
4	first_time_unit_size	N3
5	time_unit_size	N4
6	number_of_time_unit_entries	N5
7	}	

【図３９】

TimeUnitMapData()

	Syntax	Bits
1	TimeUnitMapData() {	
2	for(i=0; i<number_of_time_unit_entries; i++) {	
3	delta_time_unit_address	N6
4	}	
5	}	

【図４０】

EntryPointMapData()

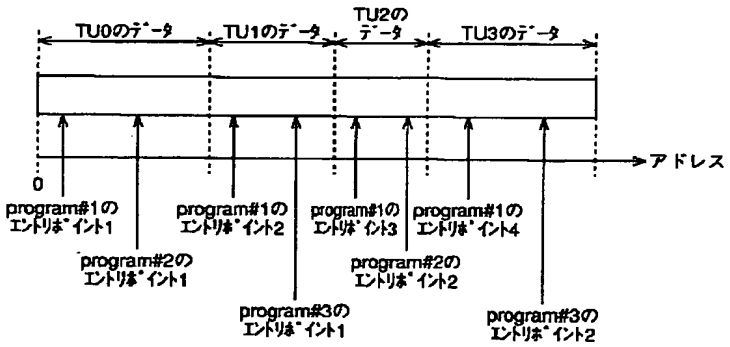
	Syntax	Bits
1	EntryPointMapData() {	
2	for(i=0; i<NUMBER_OF_ParsedPrograms; i++) {	
3	for(j=0; j<number_of_time_unit_entries; j++)	
4	entry_point_flag	1
5	entry_point_data()	
6	}	
7	}	

【図４１】

entry_point_data()

	Syntax	Bits
1	entry_point_data() {	
2	entry_point_time_stamp	K2
3	I_start_offset_from_time_unit_address	K3
4	I_end_offset_address	K4
5	P1_end_offset_address	K5
6	P2_end_offset_address	K6
7	}	

【図４２】



【図４３】

(A) program#1のEntryPointMapData

タイムユニット	entry_point_flag	entry_point_data
TU0	1	entry_point_data#1−1
TU1	1	entry_point_data#1−2
TU2	1	entry_point_data#1−3
TU3	1	entry_point_data#1−4

(B) program#2のEntryPointMapData

タイムユニット	entry_point_flag	entry_point_data
TU0	1	entry_point_data#2−1
TU1	0	−(dummy_data)
TU2	1	entry_point_data#2−2
TU3	0	−

(C) program#3のEntryPointMapData

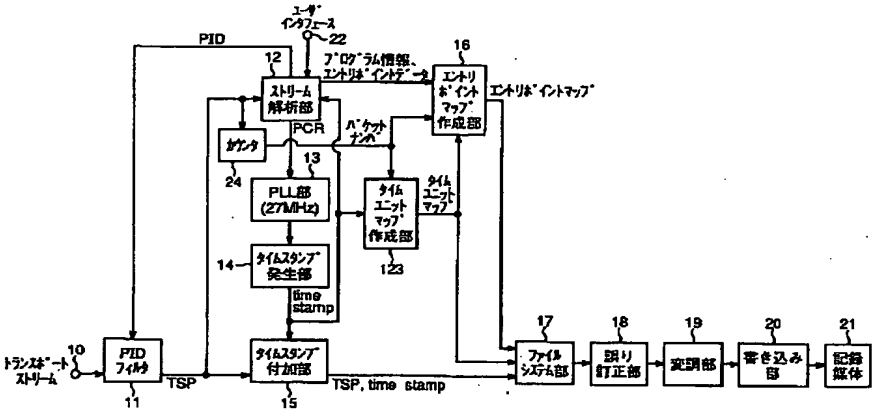
タイムユニット	entry_point_flag	entry_point_data
TU0	0	−
TU1	1	entry_point_data#3−1
TU2	0	−
TU3	1	entry_point_data#3−2

【図４４】

program#1, program#2, program#3のEntryPointMapData

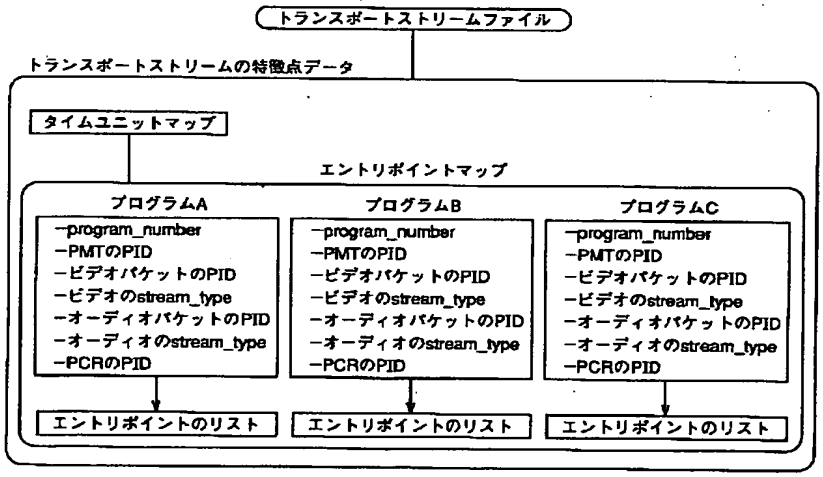
タイムユニット	Program number	entry_point_flag	entry_point_data
TU0	#1	1	entry_point_data#1−1
	#2	1	entry_point_data#2−1
	#3	0	−
TU1	#1	1	entry_point_data#1−2
	#2	0	−
	#3	1	entry_point_data#3−1
TU2	#1	1	entry_point_data#1−3
	#2	1	entry_point_data#2−2
	#3	0	−
TU3	#1	1	entry_point_data#1−4
	#2	0	−
	#3	1	entry_point_data#3−2

【図４５】

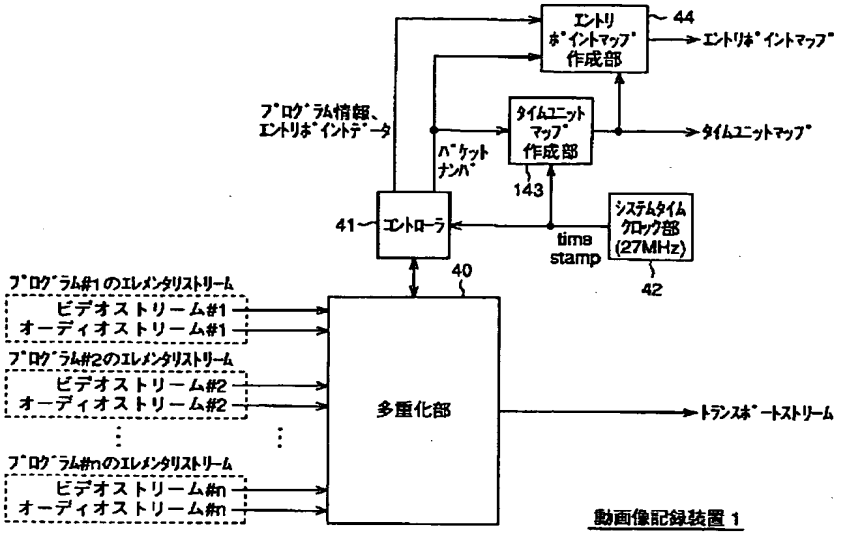


動画像記録装置１

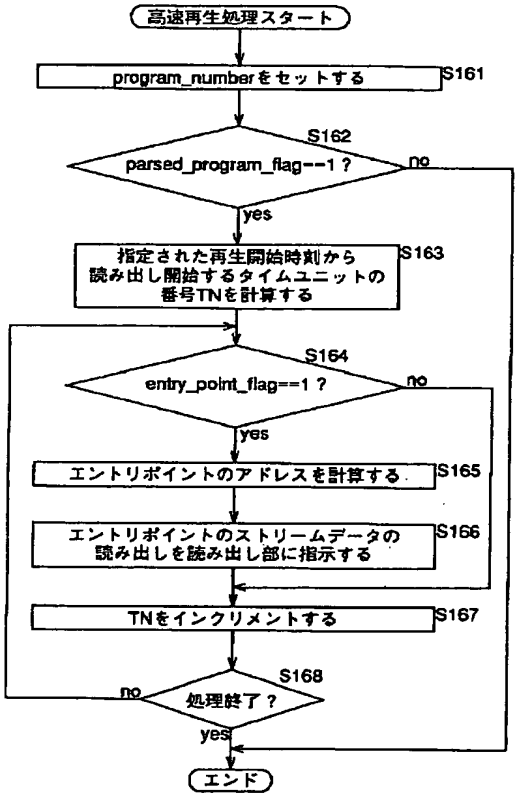
【図４６】



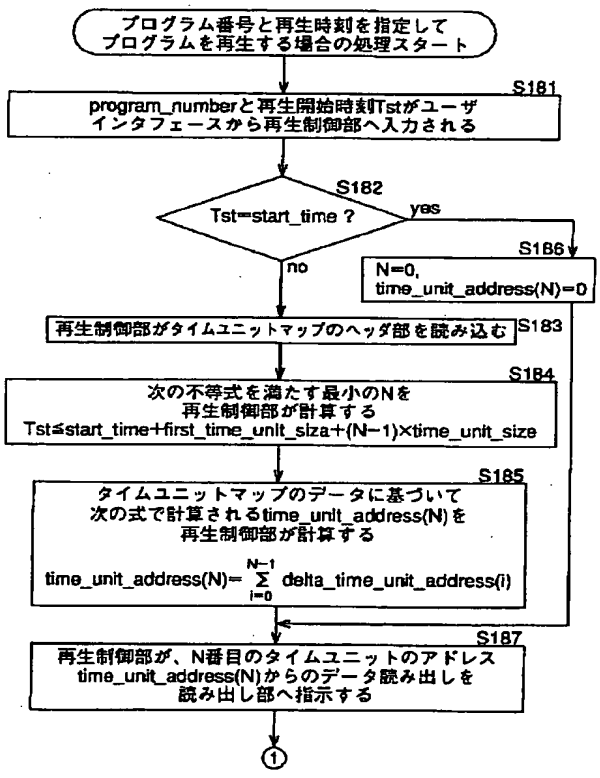
【図４７】


エントリポイントマップ作成部
44
エントリポイントマップ
プログラム情報、エントリポイントデータ
タイムユニットマップ作成部
タイムユニットマップ
パケットナンバ
143
41
エントローラ
time stamp
システムタイムクロック部(27MHz)
42
40
プログラム#1のエレメンタリストリーム
ビデオストリーム#1
オーディオストリーム#1
プログラム#2のエレメンタリストリーム
ビデオストリーム#2
オーディオストリーム#2
プログラム#nのエレメンタリストリーム
ビデオストリーム#n
オーディオストリーム#n
多重化部
トランスポートストリーム
動画像記録装置１

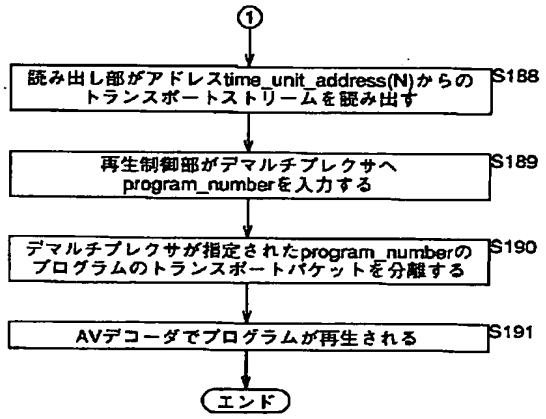
【図４８】


高速再生処理スタート
program_numberをセットする
S161
S162
parsed_program_flag==1？
no
yes
指定された再生開始時刻から読み出し開始するタイムユニットの番号TNを計算する
S163
S164
entry_point_flag==1？
no
yes
エントリポイントのアドレスを計算する
S165
エントリポイントのストリームデータの読み出しを読み出し部に指示する
S166
TNをインクリメントする
S167
S168
no
処理終了？
yes
エンド

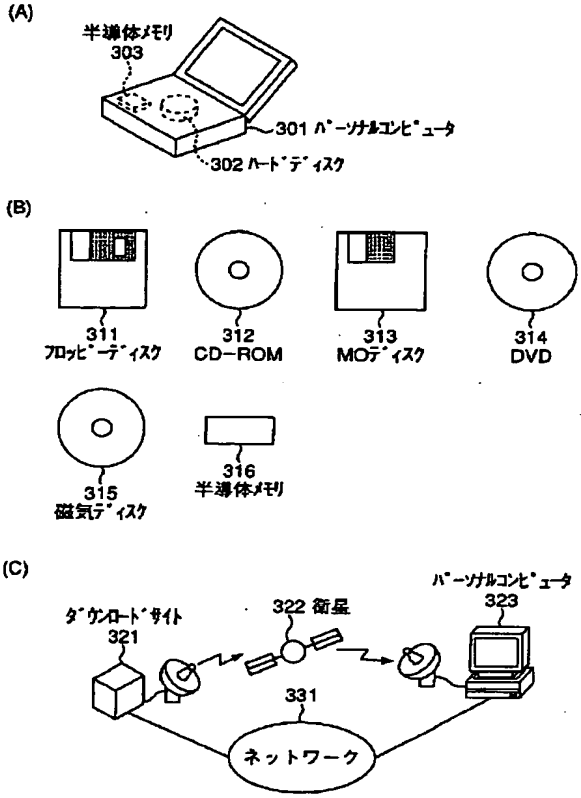
【図４９】


プログラム番号と再生時刻を指定してプログラムを再生する場合の処理スタート
S181
program_numberと再生開始時刻Tstがユーザインタフェースから再生制御部へ入力される
S182
Tst=start_time？
yes
no
S186
N=0,
time_unit_address(N)=0
再生制御部がタイムユニットマップのヘッダ部を読み込む
S183
S184
次の不等式を満たす最小のNを再生制御部が計算する
Tst≦start_time+first_time_unit_size+(N−1)×time_unit_size
S185
タイムユニットマップのデータに基づいて次の式で計算されるtime_unit_address(N)を再生制御部が計算する
time_unit_address(N)= Σ delta_time_unit_address(i)
S187
再生制御部が、N番目のタイムユニットのアドレスtime_unit_address(N)からのデータ読み出しを読み出し部へ指示する
①

【図５０】



【図５１】



フロントページの続き

(72)発明者　中村　政信
東京都品川区北品川６丁目７番35号　ソニー株式会社内

Ｆターム(参考)　5C053 FA20 FA22 FA24 GA11 GB01 GB05 GB11 GB15 GB37 GB38 HA21 JA01 JA22
5C059 KK08 MA00 PP05 PP06 PP07 RB02 RB16 RF04 SS02 SS11 SS20 SS30 UA02 UA05